Towards Interactive Language Modeling

Maartje ter Hoeve[*]
University of Amsterdam
m.a.terhoeve@uva.nl

Evgeny Kharitonov[†]
Meta AI
eugene.kharitonov@gmail.com

Dieuwke Hupkes
Meta AI
dieuwkehupkes@fb.com

Emmanuel Dupoux
Meta AI
dpx@fb.com

Abstract

Interaction between caregivers and children plays a critical role in human language acquisition and development. Therefore, it is remarkable that explicit interaction plays little to no role in artificial language modeling. In this work, we pioneer the space of interactive language modeling. We present a road map in which we detail the steps towards interactive language modeling and take the first steps on this road map, showing the initial feasibility of our approach. As such, this work aims to be the start of a larger research agenda on interactive language modeling.

1 Introduction

Interaction between children and more advanced language interlocutors (such as caregivers) plays an important role in many theories and studies on human language acquisition [4, 6]. Nonetheless, interaction plays little to no role in artificial language modeling. This is remarkable, as language modeling also has the objective to learn human language, but with artificial models. Instead, state-of-the-art language models (LMs) take large amounts of text, and need to predict the next or masked words [3, 9].

Although this setup has shown to be effective, from the perspective of human language acquisition it appears unnatural. This motivates us to investigate more interactive approaches to language modeling. An interactive approach to language modeling is not only interesting from the perspective of human language acquisition. Explicitly allowing for interaction also has the potential to make language modeling more efficient and versatile. For example, a teacher can adapt its input to a student based on the specific feedback signals it receives from the student, and a teacher that is fluent in one domain can teach the specifics of that domain to a student trained on another domain, and vice versa. Moreover, an interactive approach to language modeling has the potential to impact downstream applications, for example for foreign language teaching apps where a student can be replaced by a human.

We structure our proposal according to a teacher-student setup, in which we distinguish four main parts: (i) *the teacher*, whose role is inspired by the caregiver in the human language acquisition, (ii) *the student*, who resembles the child, (iii) *the interaction* between the teacher and the student, and (iv) *the environment* that they both share. As such, our framework combines ideas from curriculum learning [2], active learning [7] and continual learning. We detail our setup further in Section 3.

With this paper we contribute the following: (i) we define the *objective* of interactive language modeling, (ii) we present a *road map* that details the steps that need to be taken towards this objective,

[*]Work done while interning at Meta AI.
[†]Work done while at Meta AI.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

and (iii) we take the *first steps* on this road map, which show the initial feasibility of our approach. By doing so we aim to start a larger discussion and research agenda on interactive language modeling.

2 Related Work on Interactive Language Learning in NLP

Recently, a number of studies have focused on interactive language learning. Approaches on emergent communication typically focus on agents that perform certain language games, in which the agents interactively need to define a protocol to communicate [e.g., 11, 13]. Stein et al. [31] learn logical semantic representations in an interactive way. Nikolaus and Fourtassi [23] propose a proof of concept to model perception and production based learning of semantic knowledge acquisition in children. Kiseleva et al. [16, 17] take an interactive approach to language *understanding* in a recent NeurIPS challenge. To the best of our knowledge, none of the existing works have focused specifically on language modeling.

3 A Road Map towards Interactive Language Modeling

In this section we present a general road map towards interactive language modeling. We first specify our objective: *building an automated teacher-student loop for language modeling that attains good performance in the student for a fixed (low) number of bits transmitted in the interactions.*

Teachers transmit data to their students, according to a certain budget, which forces the teacher to actively choose a learning strategy, as just sending all data that is available to the teacher would not be allowed. Students have the objective to learn the language. They send a signal back that informs the teacher of their performance, e.g., a score on an exam. The interaction takes place in an environment.

In Table 1 we present a road map towards interactive language modeling. For each of the aforementioned components we detail the steps that we need to take. We also add a fifth component: the evaluation. We focus on text and acknowledge grounded interactive language modeling as an interesting future direction. Next, we take the first steps on the road map, focusing on the teacher.

4 First Steps on the Road Map

Figure 1 shows the interpretation of the student-teacher loop that we use to take the first steps on the road map. We discuss each component below. Additional implementation details are given in Appendix A.

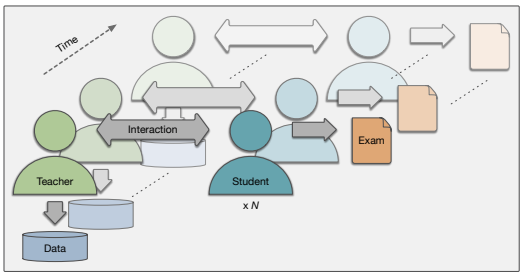


Figure 1: Teacher-student loop.

The Teacher. The teacher needs to transmit language data that will optimally help the student to learn the language. We train the teacher in a number of time steps. At each step the teacher samples data from a larger language data set according to a fixed budget. To reduce the variance in the teacher's learning process we repeat this process for multiple students, i.e., a teacher selects N "lessons" for N students. We can train multiple students simultaneously on a single GPU, avoiding a strong increase of the computational cost. The teacher is modeled as a native speaker. We represent the teacher's language understanding with a pretrained causal Transformer LM [33]. We pretrain this model on a *different* subset of the data than the teacher can select from for the students, and thus we ensure that we measure whether a teacher can teach a language as a whole, and not only a particular subset that it was trained on itself. We use REINFORCE [37] with entropy regularization [21] to learn the teacher's didactic approach.

The Student. As the teacher is the main focus of our work, we choose to keep the student side simple. We represent it as a causal Transformer LM, that we train on the data that it receives from the teacher.

The Interaction. Following Table 1, the teacher sends all selected data to the student at once. The student uses this data to train its LM and takes an exam after a predefined number of updates. The average exam score is sent back to the teacher. We use the student's last model checkpoint to compute the scores, to ensure that the learning signal for the teacher is restricted to the student's performance on the exam set, i.e., we do not expect teachers to reverse the learning process of the students.

Teacher	Student
Ways of speaking • Select data from bin;* • Generate data with own language model. **Degrees of awareness** • (No*) memory buffer of what has been sent to the student and being able to act on it (see *Interaction* cell); • (No*) explicit way of remembering what the student's fine-grained capabilities are and being able to act on it (see *Interaction* cell).	**Ways of speaking** • Generate language data in a standard LM fashion;* • Actively experiment with language generation to elicit direct feedback from the teacher (see also *Interaction* cell). **Degrees of using the teacher data** • Use all data received from the teacher; • Actively select data that is useful; • Actively know when to stop training (for example to avoid overfitting).

Interaction	Environment
Teacher side • Send all data at once;* • Send data in batches, based on student feedback (see below). Batches can be as small as single utterances, after which the student sends an utterance back, like in real human-to-human interaction (see below); • Send (mid-term) exams. **Student side** • Send a single average exam score back to the teacher;* • Send a fine-grained exam score back, e.g., – score per item on the exam set; – (average) scores of different components (tasks) of the exam(s) • Ask for feedback, for example by actively experiment with language generation for the teacher to judge ('generate own exam').	**Language** • Artificial languages, in increasing level of difficulty in terms of complexity, e.g., – random language;* – different types of structures;* – different vocabulary sizes; • Subset of human language, e.g., in terms of – semantics (e.g., different domains) – syntax (e.g., different grammatical structures) – pragmatics • Unrestricted human language. **Task** • *Teacher:* Learn to select or generate the optimal data such that the student performs well on the exam set (see cell below);* • *Teacher:* Learn to adapt to different types of students, e.g., – architectural differences – different prior knowledge (be aware of catastrophic forgetting in neural networks) • *Student:* Learn to adapt to different types of teachers (didactic strategies).

Evaluation / Exam
Teacher • Accuracy in selecting the optimal teaching protocol*
Student (Exam / Feedback for teacher) • General performance, measured in perplexity;* • Performance on specific tasks, such as – Subset of the data known to the teacher (e.g., specific domain or (grammatical) structure) – BLIMP [35]; – BIG-Bench [30]. • Scores either as an average* or more fine-grained (see *Interaction* cell).

Table 1: Road map to interactive language modeling. We detail the steps that we need to take for each of the components in the interactive language modeling setup. Steps that we take in this work are indicted by $*$.

The Environment. We design a number of artificial languages to test our approach on (see Section 5 for details). Using artificial languages is a well-tested approach to study the behavior of neural networks [1, 5, 8, 12, 14, 15, 19, 26, 27, 28, 29, 32, 36] and gives us the control we need to design our experiments in such a way that we can correctly interpret the results.

The Exam. The exam is a held-out set over which we compute the students' average perplexities. We use the negative as reward to train the teacher. We discuss further details in the next section.

5 Experiments

We test our proposed setup on a number of settings and tasks, that we describe in this section. Additional training details can be found in Appendix A.

5.1 Description of the Tasks and Baselines

Task 1 – Teaching Different Domains. For this task we design a language consisting of two strictly separated vocabularies, loosely representing two different domains in natural language. Specifically, $V_1 = \{a, b, c, d, e, f, g, h, i, j\}$, and $V_2 = \{k, l, m, n, o, p, q, r, s, t\}$. We construct sentences by randomly sampling 10 tokens from *either* of these sets. The dataset the teacher can choose from consists for 50% of V_1-sentences and for 50% of V_2-sentences. The student's exam consists of V_1-sentences only, and thus the optimal teaching strategy is to send V_1-sentences to the student.

Task 2 – Teaching Different Structures. For this task we use different sentence structures. All sentences are constructed with V_1 and are between 2 and 10 tokens long. We use two different structures: single repetitions, $(xy)^n$, and double repetitions, (xx) or $(xxyy)^n$. In the case of the single repetitions two identical tokens never occur next to each other, whereas in the case of double repetitions tokens are sampled in pairs. The data that the teacher can sample from consists for 20% of Structure 1 sentences and for 80% of Structure 2 sentences.[3] The exam set consists of sentences with Structure 1, and thus the optimal teaching strategy is to send Structure 1 sentences to the student.

Baseline Experiments. We run three baseline experiments with different didactic strategies: *oracle*, *random*, and *worst case*. Each time, we randomly select data according to the teacher budget and train a student LM with this data. For the oracle baseline we select sentences with the exam vocabulary (Task 1) or structure (Task 2). For the random baseline we randomly select sentences. For the worst case baseline all selected sentences are from a different vocabulary or structure than the exam.

6 Results

Table 2 presents the results for the baseline experiments, for the best and worst seed. Additional results per seed, including the fraction of train data that consists of the exam vocabulary or structure, are given in Appendix C.1. The results are as expected. The oracle baseline performs best, followed by the random and worst case baseline. Figure 2a and 2b show the results for Task 1 for different numbers of students per teacher. Additional plots, that explore different sentence embeddings and n-gram overlap between the train and test data are given in Appendix C.2. The teacher's didactic strategy correctly converges to the oracle baseline. For Task 2 we opted for 12 students per teacher, based on the results in Task 1. Figure 2c and 2d show the results. The teacher learns to converge to the oracle teaching strategy, although convergence is less fast than for Task 1; we do not achieve full convergence in the number of training episodes that we run these experiments for. We postulate that the two different structures are harder to distinguish, resulting in a less strong learning signal.

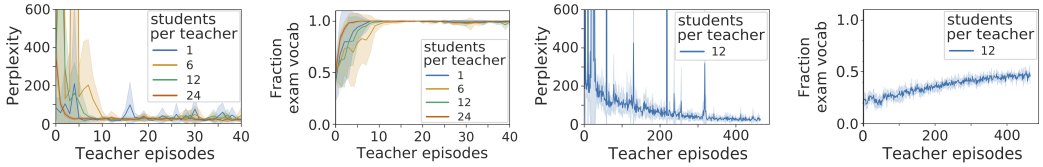
7 Conclusion

In this paper we pioneered the space of interactive language modeling, motivated by the observation that current state-of-the-art LMs are trained in a very unnatural way, from the perspective of human language acquisition. Specifically, we proposed a teacher-student loop, in which the teacher is

[3] We opt for this way of splitting the data, as we found that a student performs quite well when trained on data consisting half of Structure 1 and half of Structure 2. Having an unequal split thus allows us to make sure that we can appropriately distinguish a learned didactic approach from a random one.

Type	T1: Avg PPL Best Seed	T1: Avg PPL Worst Seed	T2: Avg PPL Best Seed	T2: Avg PPL Worst Seed
Oracle	$14.99_{\pm 5.364}$	$68.95_{\pm 87.49}$	$6.821_{\pm 0.619}$	$9.431_{\pm 3.057}$
Random	$160.9_{\pm 217.7}$	$742.5_{\pm 159.8}$	$119.0_{\pm 56.48}$	$342.1_{\pm 241.4}$
Worst Case	$4.78e4_{\pm 2.67e4}$	$8.46e4_{\pm 4.69e4}$	$299.6_{\pm 124.2}$	$595.3_{\pm 297.9}$

Table 2: Baseline results Task 1 (T1) and Task 2 (T2), for the best and worst seed. Averages and standard deviations reported based on five runs per seed.



(a) T1: Student PPL per episode. (b) T1: Fraction train data from V_1 per episode. (c) T2: Student PPL per episode. (d) T2: Fraction train data from Struct. 1 per episode.

Figure 2: Results Task 1 (T1) and Task 2 (T2). Results on exam data, reported as average and standard deviation over five random seeds.

inspired by the caregiver and the student resembles the child in the human language acquisition. We presented a road map that details the steps towards interactive language modeling for each of the components of the teacher-student loop. We led by example and took the first steps on this road map, leading to a tangible proof of concept of our proposal. As such, we structured the space of interactive language modeling and aim to inspire a larger research agenda on interactive language modeling.

References

[1] J. Batali. Artificial evolution of syntactic aptitude. In *Proceedings from the Sixteenth Annual Conference of the Cognitive Science Society*, pages 27–32. Lawrence Erlbaum Associates Hillsdale, NJ, 1994.

[2] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In A. P. Danyluk, L. Bottou, and M. L. Littman, editors, *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pages 41–48. ACM, 2009. doi: 10.1145/1553374.1553380. URL `https://doi.org/10.1145/1553374.1553380`.

[3] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html`.

[4] J. Bruner. Child's talk: Learning to use language. *Child Language Teaching and Therapy*, 1(1): 111–114, 1985.

[5] R. Chaabouni, R. Dessì, and E. Kharitonov. Can transformers jump around right in natural language? assessing performance transfer from SCAN. In J. Bastings, Y. Belinkov, E. Dupoux, M. Giulianelli, D. Hupkes, Y. Pinter, and H. Sajjad, editors, *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2021, Punta Cana, Dominican Republic, November 11, 2021*, pages 136–148. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.blackboxnlp-1.9. URL `https://doi.org/10.18653/v1/2021.blackboxnlp-1.9`.

[6] E. V. Clark. Conversation and language acquisition: A pragmatic approach. *Language Learning and Development*, 14(3):170–185, 2018.

[7] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *J. Artif. Intell. Res.*, 4:129–145, 1996. doi: 10.1613/jair.295. URL https://doi.org/10.1613/jair.295.

[8] G. Dagan, D. Hupkes, and E. Bruni. Co-evolution of language and agents in referential games. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2993–3004, Online, Apr. 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.eacl-main.260.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[10] Y. Dong, Y. Shen, E. Crawford, H. van Hoof, and J. C. K. Cheung. Banditsum: Extractive summarization as a contextual bandit. *arXiv preprint arXiv:1809.09672*, 2018.

[11] K. Evtimova, A. Drozdov, D. Kiela, and K. Cho. Emergent communication in a multi-modal, multi-step referential game. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJGZq6g0-.

[12] F. A. Gers and J. Schmidhuber. LSTM recurrent networks learn simple context-free and context-sensitive languages. *IEEE Trans. Neural Networks*, 12(6):1333–1340, 2001. doi: 10.1109/72.963769. URL https://doi.org/10.1109/72.963769.

[13] L. Harding Graesser, K. Cho, and D. Kiela. Emergent linguistic phenomena in multi-agent communication games. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3700–3710, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1384. URL https://aclanthology.org/D19-1384.

[14] D. Hupkes, S. Veldhoen, and W. H. Zuidema. Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *J. Artif. Intell. Res.*, 61:907–926, 2018. doi: 10.1613/jair.1.11196. URL https://doi.org/10.1613/jair.1.11196.

[15] D. Hupkes, V. Dankers, M. Mul, and E. Bruni. Compositionality decomposed: how do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795, 2020.

[16] J. Kiseleva, Z. Li, M. Aliannejadi, S. Mohanty, M. ter Hoeve, M. S. Burtsev, A. Skrynnik, A. Zholus, A. Panov, K. Srinet, A. Szlam, Y. Sun, K. Hofmann, M. Côté, A. H. Awadallah, L. Abdrazakov, I. Churin, P. Manggala, K. Naszádi, M. van der Meer, and T. Kim. Interactive grounded language understanding in a collaborative environment: IGLU 2021. In D. Kiela, M. Ciccone, and B. Caputo, editors, *NeurIPS 2021 Competitions and Demonstrations Track, 6-14 December 2021, Online*, volume 176 of *Proceedings of Machine Learning Research*, pages 146–161. PMLR, 2021. URL https://proceedings.mlr.press/v176/kiseleva22a.html.

[17] J. Kiseleva, A. Skrynnik, A. Zholus, S. Mohanty, N. Arabzadeh, M. Côté, M. Aliannejadi, M. Teruel, Z. Li, M. S. Burtsev, M. ter Hoeve, Z. Volovikova, A. I. Panov, Y. Sun, K. Srinet, A. Szlam, and A. H. Awadallah. IGLU 2022: Interactive grounded language understanding in a collaborative environment at neurips 2022. *CoRR*, abs/2205.13771, 2022. doi: 10.48550/arXiv.2205.13771. URL https://doi.org/10.48550/arXiv.2205.13771.

[18] W. Kool, H. van Hoof, and M. Welling. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 3499–3508. PMLR, 2019. URL http://proceedings.mlr.press/v97/kool19a.html.

[19] B. M. Lake and M. Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In J. G. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2879–2888. PMLR, 2018. URL http://proceedings.mlr.press/v80/lake18a.html.

[20] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[21] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In M. Balcan and K. Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1928–1937. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/mniha16.html.

[22] S. Narayan, S. B. Cohen, and M. Lapata. Ranking sentences for extractive summarization with reinforcement learning. *arXiv preprint arXiv:1802.08636*, 2018.

[23] M. Nikolaus and A. Fourtassi. Modeling the interaction between perception-based and production-based learning in children's early acquisition of semantic knowledge. 2021.

[24] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

[25] J. Rapin and O. Teytaud. Nevergrad - A gradient-free optimization platform. https://GitHub.com/FacebookResearch/Nevergrad, 2018.

[26] P. Rodriguez. Simple recurrent networks learn context-free and context-sensitive languages by counting. *Neural computation*, 13(9):2093–118, 2001.

[27] P. Rodriguez, J. Wiles, and J. L. Elman. A recurrent neural network that learns to count. *Connection Science*, 11(1):5–40, 1999.

[28] D. Rodríguez Luna, E. M. Ponti, D. Hupkes, and E. Bruni. Internal and external pressures on language emergence: least effort, object constancy and frequency. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4428–4437, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.397. URL https://aclanthology.org/2020.findings-emnlp.397.

[29] D. Saxton, E. Grefenstette, F. Hill, and P. Kohli. Analysing mathematical reasoning abilities of neural models. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=H1gR5iR5FX.

[30] A. Srivastava, A. Rastogi, A. Rao, A. A. M. Shoeb, A. Abid, A. Fisch, A. R. Brown, A. Santoro, A. Gupta, A. Garriga-Alonso, A. Kluska, A. Lewkowycz, A. Agarwal, A. Power, A. Ray, A. Warstadt, A. W. Kocurek, A. Safaya, A. Tazarv, A. Xiang, A. Parrish, A. Nie, A. Hussain, A. Askell, A. Dsouza, A. Rahane, A. S. Iyer, A. Andreassen, A. Santilli, A. Stuhlmüller, A. M. Dai, A. La, A. K. Lampinen, A. Zou, A. Jiang, A. Chen, A. Vuong, A. Gupta, A. Gottardi, A. Norelli, A. Venkatesh, A. Gholamidavoodi, A. Tabassum, A. Menezes, A. Kirubarajan, A. Mullokandov, A. Sabharwal, A. Herrick, A. Efrat, A. Erdem, A. Karakas, and et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *CoRR*, abs/2206.04615, 2022. doi: 10.48550/arXiv.2206.04615. URL https://doi.org/10.48550/arXiv.2206.04615.

[31] K. Stein, L. Harter, and L. Geiger. Shapelurn: An interactive language learning game with logical inference. In *Proceedings of the First Workshop on Interactive Learning for Natural Language Processing*, pages 16–24, 2021.

[32] O. van der Wal, S. de Boer, E. Bruni, and D. Hupkes. The grammar of emergent languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3339–3359, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.270. URL https://aclanthology.org/2020.emnlp-main.270.

[33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[34] T. Vieira. Gumbel-max trick and weighted reservoir sampling, 2014. URL http://timvieira.github.io/blog/post/2014/08/01/gumbel-max-trick-and-weighted-reservoir-sampling/.

[35] A. Warstadt, A. Parrish, H. Liu, A. Mohananey, W. Peng, S.-F. Wang, and S. R. Bowman. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392, 2020.

[36] J. Wiles and J. Elman. Learning to count without a counter: A case study of dynamics and activation landscapes in recurrent networks. In *Proceedings of the seventeenth annual conference of the cognitive science society*, number s 482, page 487. Erlbaum Hillsdale, NJ, 1995.

[37] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Additional Implementation Details

A.1 Data Selection by the Teacher

We use REINFORCE [37] with entropy regularization [21] to learn the teacher's didactic approach. We also experimented with gradient-free optimization approaches such as the ones implemented in Nevergrad [25], but found REINFORCE to be more flexible in our case and therefore a better fit for our needs. We want to optimize the teacher's policy such that it learns to select the optimal data to train the student on, given a predefined budget. The policy is a one-layer feed forward neural network, that outputs a score for each sentence, i.e., the teacher's policy network takes a sentence embedding as input, based on the pretrained Transformer LM that we use to represent the teacher's language understanding. An action is modeled as selecting k sentences from the larger data set, where k is a predefined teacher budget. We use the GumbelTopK trick [34, 18] to sample k sentences without replacement, based on the teacher policy's output scores. We compute the log probabilities (needed to compute the loss) for each sample by adding the log probabilities of each element in the sample. We explain the rationale behind this in Appendix B.

A.2 Training Details Task 1 and Task 2

The teacher LM is trained on 100 unique sentences till convergence. The dataset the teacher can sample from for the student consists of 100 different unique sentences. The exam consists of 10 unique sentences and we set the teacher budget to 10 as well. We run our experiments with five different random seeds and report the averages and standard deviations. We use the negative perplexity of the student on the exam as reward for the teacher. We experiment with two different sentence embeddings for the teacher: average word embeddings and the average of the last hidden layer. We train students for a predefined number of steps that we determine by inspecting the loss and perplexity curves of training an LM once before the actual experiments. We base the threshold on when a student LM starts to overfit, so that a teacher can get clear feedback signals. We set this value to 400 for Task 1 and 300 for Task 2. Automatically determining when the students stops training is an important avenue for future work (Table 1). We use Fairseq's [24] `transformer_lm`[4] for the implementation of the Transformer LMs. We use up to four GPUs with 32 GB RAM per experiment. The exact number depends on the number of students per teacher, as we can fit up to 6 students on a single GPU due to our multiprocessing implementation.

A.3 Additional Details Baseline Experiments

In each experiment, we randomly select data according to the teacher budget. We do this five times and each time train a student LM with the selected data.

B Computing the Probability of a Top-K Sample

Our objective is to find the (log) probability of sampling the subset $(i_1, ...i_K)$ from $\{1, ..., N\}$ *without* replacement from the categorical probability $(p_1, ..., p_N)$.

Let us first consider sampling K elements from the $\{1, ..., N\}$ *with* replacement. In that case

[4] `https://fairseq.readthedocs.io/en/latest/command_line_tools.html`

$$p(i_1, ..., i_K) = \prod_{k=1}^{K} p_{i_k}. \tag{1}$$

If we allow for all possible permutations of observing $(i_1, ..., i_K)$ we get

$$p(i_1, ..., i_K) = C \prod_{k=1}^{K} p_{i_k}, \tag{2}$$

where $C = K!$.

To go from sampling *with* replacement, to sampling *without* replacement, we consider event $A =$ "all sampled elements $(i_1, ..., i_K)$ are unique". Then

$$\begin{aligned} &p_{\text{w/o replacement}}(i_1, ..., i_K) = \\ &p_{\text{w/ replacement}}(i_1, ..., i_K | A). \end{aligned} \tag{3}$$

Applying Bayes Rule gives us:

$$\begin{aligned} &p_{\text{w/o replacement}}(i_1, ..., i_K) = \\ &\frac{p_{\text{w/ replacement}}(A | i_1, ..., i_K) p_{\text{w/ replacement}}(i_1, ..., i_K)}{p_{\text{w/ replacement}}(A)}. \end{aligned} \tag{4}$$

As in our case all samples in $(i_1, ..., i_K)$ are unique we know that

$$p_{\text{w/ replacement}}(A | i_1, ..., i_K) = 1. \tag{5}$$

Combining this with Equation 2 gives us

$$p_{\text{w/o replacement}}(i_1, ..., i_K) = \frac{C \prod_{k=1}^{K} p_{i_k}}{p(A)}, \tag{6}$$

and thus

$$p_{\text{w/o replacement}}(i_1, ..., i_K) \propto \prod_{k=1}^{K} p_{i_k}, \tag{7}$$

and

$$\log p_{\text{w/o replacement}}(i_1, ..., i_K) \propto \sum_{k=1}^{K} \log p_{i_k}. \tag{8}$$

From an implementation perspective this boils down to the following steps:

1. We compute the scores per sentence.
2. We sample K sentences without replacement, using the GumbelTopK trick.
3. We compute the log probabilities for each score: $\log \text{softmax}(scores)$.
4. We compute the log probability of our sample by adding the log probabilities of the elements in our sample, according to Equation 8.

B.1 Comparison to Prior Work

Our problem of sampling K sentences as a single action is similar to the problem formulation of using reinforcement learning for extractive summarization to optimize for Rouge [20] directly. In this setting K sentences need to be selected from a document. This results in a very large search space. Narayan et al. [22] limit the search space by first selecting n sentences that have a high Rouge score. Then all possible summaries are made with these n sentences. These summaries are ranked according to their Rouge scores and the top K sentences are taken as action. This approach has the disadvantage that it limits the search space heuristically, which does not guarantee that the best summary is found. Dong et al. [10] frame the problem as a contextual bandit problem, which allows them to sample from the true action space. We choose our approach as it is intuitive, simple and effective.

C Additional Results

C.1 Additional Results Baseline Experiments Task 1 and Task 2

In Table 3 we present the results for our baseline runs on all five seeds for Task 1, and in Table 4 we present the results for our baseline runs on all five seeds for Task 2.

Baseline	Seed	Avg Perplexity	Avg train from test	Avg unigram overlap	Avg bigram overlap	Avg trigram overlap
Random	6639	$193.9_{\pm 100.3}$	$0.46_{\pm 0.14}$	$0.46_{\pm 0.14}$	$0.278_{\pm 0.07}$	$0.023_{\pm 0.009}$
	7519	$683.1_{\pm 634.3}$	$0.52_{\pm 0.15}$	$0.52_{\pm 0.15}$	$0.291_{\pm 0.10}$	$0.030_{\pm 0.010}$
	1007	$742.5_{\pm 159.8}$	$0.50_{\pm 0.17}$	$0.50_{\pm 0.17}$	$0.298_{\pm 0.10}$	$0.035_{\pm 0.014}$
	4520	$160.9_{\pm 217.7}$	$0.54_{\pm 0.16}$	$0.54_{\pm 0.16}$	$0.327_{\pm 0.09}$	$0.035_{\pm 0.025}$
	4527	$307.1_{\pm 295.1}$	$0.58_{\pm 0.17}$	$0.58_{\pm 0.17}$	$0.349_{\pm 0.10}$	$0.035_{\pm 0.014}$
Oracle	6639	$14.99_{\pm 5.364}$	$1.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.551_{\pm 0.06}$	$0.072_{\pm 0.029}$
	7519	$44.37_{\pm 58.94}$	$1.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.611_{\pm 0.02}$	$0.085_{\pm 0.017}$
	1007	$68.95_{\pm 87.49}$	$1.00_{\pm 0.00}$	$1.00_{pm 0.00}$	$0.598_{\pm 0.02}$	$0.077_{\pm 0.025}$
	4520	$15.65_{\pm 4.616}$	$1.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.578_{\pm 0.02}$	$0.087_{\pm 0.028}$
	4527	$23.66_{\pm 21.44}$	$1.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.624_{\pm 0.02}$	$0.095_{\pm 0.019}$
Worst case	6639	$8.46e4_{\pm 4.69e4}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$
	7519	$7.03e4_{\pm 3.73e4}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$
	1007	$8.17e4_{\pm 4.26e4}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$
	4520	$4.78e4_{\pm 2.67e4}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$
	4527	$6.69e4_{\pm 1.98e4}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$

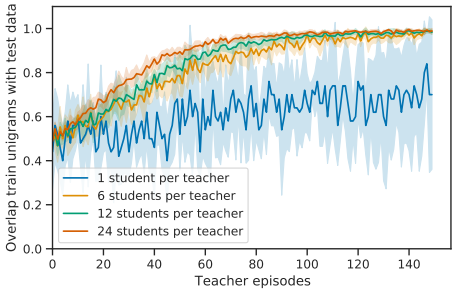
Table 3: Baseline results for Task 1. Different domains. Averages and standard deviations reported based on five runs per seed.

C.2 Additional Results Task 1 and Task 2

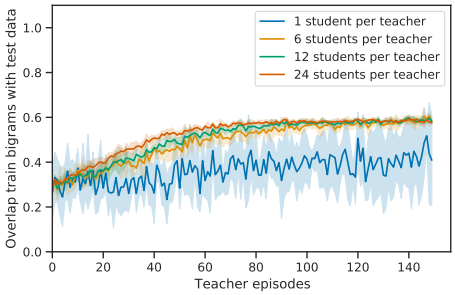
In this section we present the plots for the n-gram overlap for Task 1 in Figures 3 and 4. We present the plots for the n-gram overlap for Task 2 in Figure 5.

Baseline	Seed	Avg Perplexity	Avg train from test	Avg unigram overlap	Avg bigram overlap	Avg trigram overlap
Random	6639	$119.0_{\pm 56.48}$	$0.18_{\pm 0.04}$	$1.00_{\pm 0.00}$	$0.401_{\pm 0.033}$	$0.030_{\pm 0.020}$
	7519	$162.8_{\pm 201.9}$	$0.24_{\pm 0.05}$	$1.00_{\pm 0.00}$	$0.408_{\pm 0.044}$	$0.035_{\pm 0.038}$
	1007	$234.1_{\pm 192.0}$	$0.24_{\pm 0.12}$	$1.00_{\pm 0.00}$	$0.414_{\pm 0.034}$	$0.034_{\pm 0.020}$
	4520	$161.7_{\pm 190.6}$	$0.22_{\pm 0.04}$	$1.00_{\pm 0.00}$	$0.410_{\pm 0.023}$	$0.038_{\pm 0.033}$
	4527	$342.1_{\pm 241.4}$	$0.12_{\pm 0.08}$	$1.00_{\pm 0.00}$	$0.348_{\pm 0.024}$	$0.013_{\pm 0.017}$
Oracle	6639	$6.973_{\pm 1.534}$	$1.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.720_{\pm 0.044}$	$0.151_{\pm 0.022}$
	7519	$7.626_{\pm 2.298}$	$1.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.682_{\pm 0.056}$	$0.177_{\pm 0.033}$
	1007	$7.895_{\pm 1.106}$	$1.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.726_{\pm 0.045}$	$0.207_{\pm 0.025}$
	4520	$6.821_{\pm 0.619}$	$1.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.740_{\pm 0.073}$	$0.197_{\pm 0.054}$
	4527	$9.431_{\pm 3.057}$	$1.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.700_{\pm 0.056}$	$0.174_{\pm 0.017}$
Worst case	6639	$595.3_{\pm 297.9}$	$0.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.326_{\pm 0.026}$	$0.00_{\pm 0.00}$
	7519	$317.2_{\pm 235.8}$	$0.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.311_{\pm 0.018}$	$0.00_{\pm 0.00}$
	1007	$508.1_{\pm 155.7}$	$0.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.345_{\pm 0.017}$	$0.00_{\pm 0.00}$
	4520	$299.6_{\pm 124.2}$	$0.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.310_{\pm 0.027}$	$0.00_{\pm 0.00}$
	4527	$432.8_{\pm 72.05}$	$0.00_{\pm 0.00}$	$1.00_{\pm 0.00}$	$0.330_{\pm 0.035}$	$0.00_{\pm 0.00}$

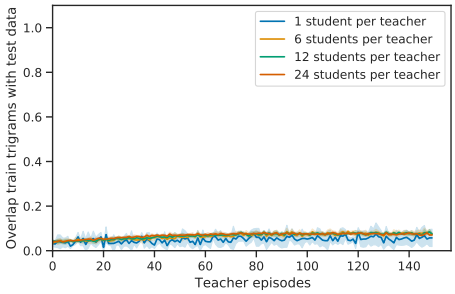
Table 4: Baseline results for Task 2. Different structures. Averages and standard deviations reported based on five runs per seed.



(a) Unigram overlap between train and test data.

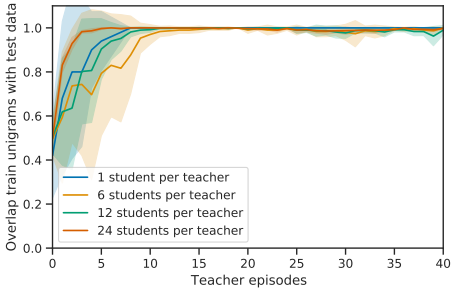


(b) Bigram overlap between train and test data.

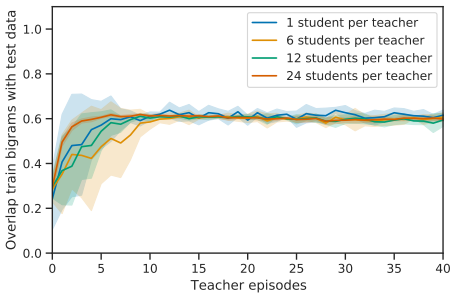


(c) Trigram overlap between train and test data.

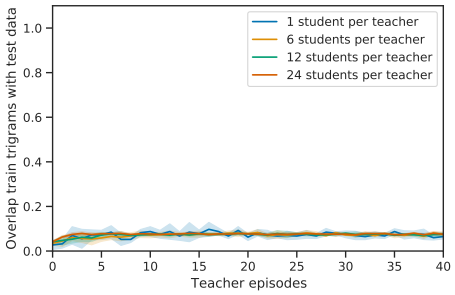
Figure 3: Additional results Task 1 – Different domains. Plots for different numbers of students per teacher. Results per setting reported as average and standard deviation over five random seeds. Average word embedding as sentence embeddings.



(a) Unigram overlap between train and test data.

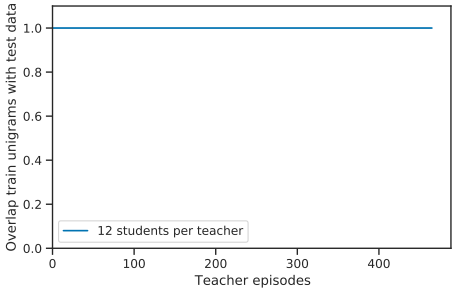


(b) Bigram overlap between train and test data.

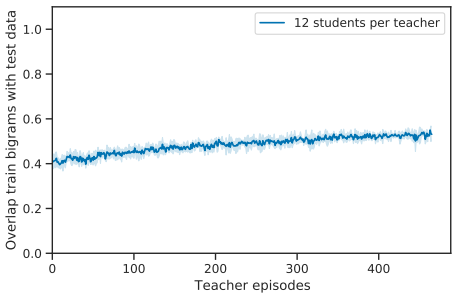


(c) Trigram overlap between train and test data.

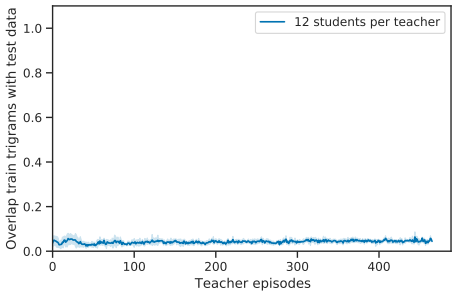
Figure 4: Additional results Task 1 – Different domains. Plots for different numbers of students per teacher. Results per setting reported as average and standard deviation over five random seeds. Average hidden layer embedding as sentence embeddings.



(a) Unigram overlap between train and test data.



(b) Bigram overlap between train and test data.



(c) Trigram overlap between train and test data.

Figure 5: Additional results Task 2 – Different structures. Results per setting reported as average and standard deviation over five random seeds.