Does the Order of Training Samples Matter? Improving Neural Data-to-Text Generation with Curriculum Learning

Ernie Chang, Hui-Syuan Yeh, Vera Demberg
Dept. of Language Science and Technology, Saarland University
{cychang,yehhui,vera}@coli.uni-saarland.de

Abstract

Recent advancements in data-to-text generation largely take on the form of neural end-to-end systems. Efforts have been dedicated to improving text generation systems by changing the order of training samples in a process known as *curriculum learning*. Past research on sequence-to-sequence learning showed that *curriculum learning* helps to improve both the performance and convergence speed. In this work, we delve into the same idea surrounding the training samples consisting of structured data and text pairs, where at each update, the curriculum framework selects training samples based on the model's competence. Specifically, we experiment with various difficulty metrics and put forward a *soft edit distance metric* for ranking training samples. Our benchmarks show faster convergence speed where training time is reduced by 38.7% and performance is boosted by 4.84 BLEU.

1 Introduction

Neural data-to-text generation has been the subject of much recent research. The task aims at transforming source-side structured data into target-side natural language descriptions (Reiter and Dale, 2000; Barzilay and Lapata, 2005). The process typically involves mini-matches which are randomly sampled with a fixed size from the training set to feed into the model at each training step. In this paper, we apply curriculum learning to this process, which was explored in neural machine translation (Platanios et al., 2019; Zhou et al., 2020), and show how it can help in neural data-to-text generation.

The main idea in curriculum learning is to present the training data in a specific order, starting from *easy* examples and moving on to more *difficult* ones, as the learner becomes more *competent*. When starting out with easier instances, the risk of getting stuck in local optima early on in training is reduced, since the loss functions in neural models are typically highly non-convex (Bengio et al., 2009). This learning paradigm enables flexible batch configurations by considering the material properties as well as the state of the learner. The idea brings in two potential benefits: (1) It speeds up the convergence and reduces the computational cost. (2) It boosts the model performance, without having to change the model or add data.

With the release of large data-to-text datasets (e.g. Wikibio (Lebret et al., 2016), Totto (Parikh et al., 2020), E2E (Novikova et al., 2017)), neural data-to-text generation is now at a point where training speed and the order of samples may begin to make a real difference. We here show the efficacy of curriculum learning with a general LSTM-based sequence-to-sequence model and define *difficulty metrics* that can assess the training instances, using a sucessful *competence function* which estimates the model capability during training. Such metrics have not yet been explored in neural data-to-text generation.

In this paper, we explore the effectiveness of various difficulty metrics and propose a soft edit distance metric, which leads to substantial improvements over other metrics. Crucially, we observe that difficulty metrics that consider data-text samples jointly lead to stronger improvements than metrics that consider text or data samples alone. In summary, this work makes the following contributions towards neural data-to-text generation:

1. We show that by simply changing the order of samples during training, neural models can be improved via the use of curriculum learning.

2. We explore various *difficulty metrics* at the level of the data, text, and data-text pairs, and propose an effective novel metric.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 727–733
April 19 - 23, 2021.

2 Related work

The idea of teaching algorithms in a similar manner as humans, incrementally from easy concepts to more difficult ones dates back to *incremental learning*, which was discussed in light of theories of cognitive development relating to the processes of acquisition in young children (Elman, 1993; Krueger and Dayan, 2009; Plunkett and Marchman, 1993). Bengio et al. (2009) first demonstrated empirically that curriculum learning approaches can decrease training times and improve generalization; later approach address these issues by changing the mini-batch sampling strategy to also include model competence (Kocmi and Bojar, 2017; Zhou et al., 2020; Platanios et al., 2019; Liu et al., 2020; Zhang et al., 2018, 2019). While sample difficulty can be assessed for text samples and data samples or jointly, various measures have been proposed for text samples including n-gram frequency Haffari (2009); Platanios et al. (2019), token rarity, and sentence length (Liu et al., 2020; Platanios et al., 2019). Our approach considers data and text jointly, similar to edit distance metric – Levenshtein (Levenshtein, 1966) and Damerau-Levenshtein Distance (Damerau, 1964; Brill and Moore, 2000a), which was used as a content ordering metric in Wiseman et al. (2017) to measure the extent of alignment between data slots and text tokens.

3 Preliminaries of Curriculum Learning

We base our curriculum learning framework on the two standard components: (1) model competence (how capable the current model is at time t), and (2) sample difficulty, which makes independent judgement on each sample's difficulty. Specifically, we adopt the competence function $c(t)$ for a model at time t as in Platanios et al. (2019); Liu et al. (2020):

$$c_{\text{sqrt}}(t) \in (0, 1] = \min\left(1, \sqrt{t\frac{1-c_0^2}{\lambda_t} + c_0^2}\right). \quad (1)$$

Where λ_t is a hyperparameter defining the length of the curriculum and is set to 2.5 as in Liu et al. (2020). $c_0 = 0.1$ as Platanios et al. (2019). Following this formulation, the number of new training examples per unit time is reduced as training progresses to give the learner sufficient time to obtain new knowledge. The sequence-to-sequence model learns using the curriculum as outlined in Algorithm 1 by primarily making batch-wise decisions about which samples to add to each batch. This decision is determined by comparing the *competence* score with the *difficulty* score as shown in Algorithm 1.

Algorithm 1: Curriculum Learning Algorithm

Input: Training set, $\mathcal{D} = \{s_d, s_t\}_{i=1}^{M}$, consisting of M samples, model ($\mathcal{T}$), difficulty metric (d), and competence function (c).

1 Compute the difficulty, $d(s_i)$, for each data-text pair $\in \mathcal{D}$ (Section 4).
2 Compute the CDF score $\bar{d}(s_i)$ of $d(s_i)$, where $\bar{d}(s_i) \in [0, 1]$ (See Figure 2).
3 **for** *training step* $t = 1, \ldots$ **do**
4 Compute the model competence $c(t)$ with $\mathcal{T}$.
5 Train $\mathcal{T}$ on sampled data batch, B_t, drawn uniformly from all $s_i \in \mathcal{D}$, such that $\bar{d}(s_i) \leq c(t)$.
6 **if** $c(t) = 1$ **then**
7 break.

Output: Trained model.

4 Difficulty Metrics

For ease of discussion, we denote sequence to be s, which can be either data or text, or their concatenation. For comparison, the difficulty metrics use the unit tokens as tokenized by SpaCy[1]. We begin with discussion on *length* and *word rarity*, which were previously applied by Kocmi and Bojar (2017); Platanios et al. (2019) on text sentences.

Length. Length-based difficulty is based on the intuition that longer sequences are harder to encode, and that early errors may propagate during the decoding process, making longer sentences also harder to generate. It is defined as:

$$d_{\text{length}}(s) = N. \quad (2)$$

Rarity. *Word rarity* of a sentence is defined as the product of the unigram probabilities (Platanios et al., 2019). This metric implicitly incorporates information about sentence length since longer sentence scores are sum of more terms and are thus likely to be larger. The difficulty metric for word rarity of a sequence s is defined as:

$$d_{\text{rarity}}(s) = -\sum_{k=1}^{N} \log p(w_k). \quad (3)$$

[1] https://spacy.io/api/tokenizer

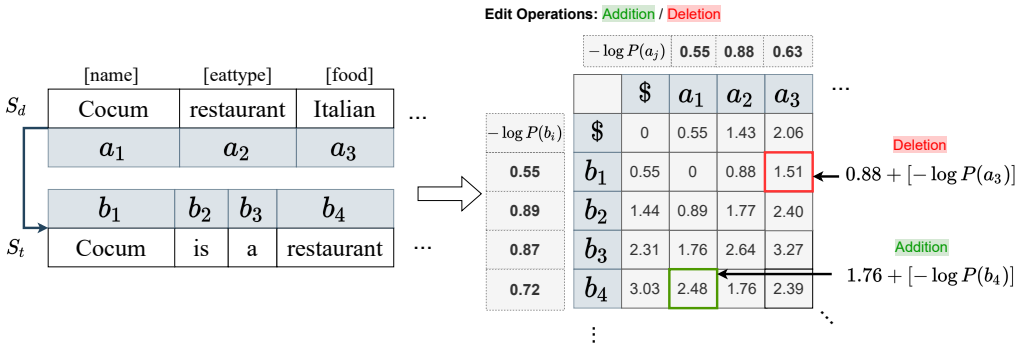


Figure 1: Depiction of the process of *soft edit distance metric* with the *Wagner-Fischer table*. Each cell in the table represents the edit distance to convert data substring's into the text substring.

Damerau-Levenshtein Distance. To consider data and text jointly, we measure the alignment between data slots and text using *the Damerau-Levenshtein Distance* (d_{dld}) (Brill and Moore, 2000a).

We calculate the minimum number of edit operations needed to transform data (s_d) into text (s_t)[2], and relies only four operations: (a) substitute a word in s_d to a different word, (b) insert a word into s_d, (c) delete a word of s_d, and (d) transpose two adjacent words of s_d. The process involves recursive calls that compute distance between substrings $s_d^i \in s_d$ and $s_t^i \in s_t$ at i^{th} comparison.

Soft Data-to-Text Edit Distance. We here present the proposed *soft edit distance (SED)*: (1) We include the basic *add* and *delete* edit operations as in the Levenshtein Distance (Levenshtein, 1966), which was used in Levenshtein Transformer (Gu et al., 2019) as the only two necessary operations for decoding sequences since it correlates well with human text writing where humans "*can revise, replace, revoke or delete any part of their generated text*". We call this variant the plain edit distance *(PED)*. (2) Next, we weight the indicator function $\mathbb{1}(s_d^i, s_t^i)$ for each edit operation with the negative logarithmic unigram probability $-\log p(w)$ for each token $w \in s^i_{\{t|d\}}$, in order to incorporate the idea of *word rarity* into the edit distance metric. For the *delete* operation, we use the $w \in s_d^i$ and for *add* operations, we use $w \in s_t^i$. This is unlike the previous proposal by Brill and Moore (2000b), in which edits are weighted by the token transition probabilities – this is not suitable for our scenario because there is no natural order of the slot sequence in data samples.

[2]Previous work applies *the Damerau-Levenshtein Distance* to slots in data (e.g. "[name]" in Figure 5) and extracts slots from text.

The soft distance metric d_{sed} is in principle similar to calculating the logarithmic sum as defined in the *rarity* function, but instead incrementally compares all substrings and calculates their edit distances. This way, d_{sed} includes the information on *length*, *rarity* but also combining the edit operations. We show this process in Figure 5.

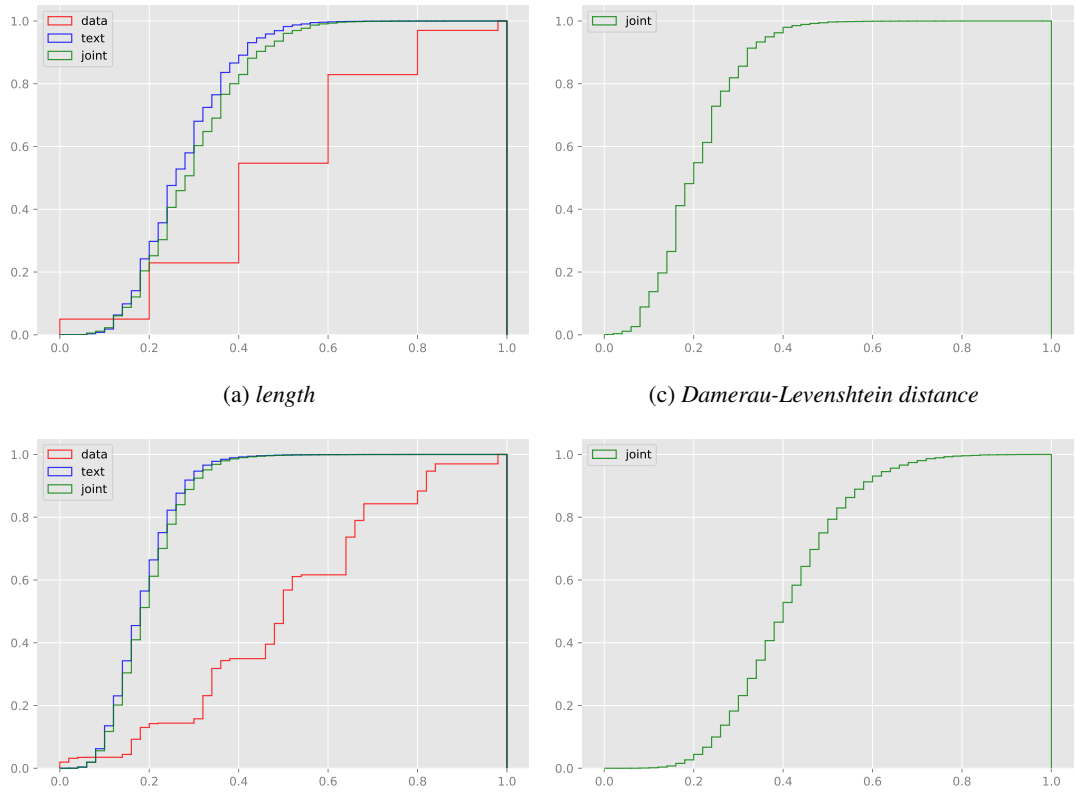
Note that we can compute *length* and *rarity* on the concatenation of input data and text sequence, or as individual sequences; whereas *Damerau-Levenshtein distance* and *soft edit distance* are computed jointly on data and text.

5 Experiment Setting

Data. We conduct experiments on the E2E (Novikova et al., 2017) and WebNLG (Colin et al., 2016) datasets. E2E is a crowd-sourced dataset containing 50k instances in the restaurant domain. The inputs are dialogue acts consisting of three to 8 slot-value pairs. WebNLG contains 25k instances describing entities belonging to 15 distinct DBpedia categories, where the data contain are up to 7 RDF triples of the form *(subject, relation, object)*.

Configurations. The LSTM-based model is implemented based on PyTorch (Paszke et al., 2019). We use 200-dimensional token embeddings and the Adam optimizer with an initial learning rate at 0.0001. Batch size is kept at 28, and we decode with beam search with size 5. The performance scores are averaged over 5 random initialization runs.

Settings. We first perform ablation studies (Table 2) on the impact of difficulty metrics on data, text or both (joint). We also analyse the average bin size for each metric – a metric that gives the same score to many instances creates large bins.



(a) *length* (c) *Damerau-Levenshtein distance*

(b) *rarity* (d) *soft edit distance*

Figure 2: The histogram of the cumulative density function for difficulty metrics.

This means that the order of samples within the bin will still be random. On the other hand, a metric that assigns a lot of different difficulty scores to the instances can yield a more complete ordering (and a smaller step size in moving from one level of difficulty to the next). We present the change in performance (BLEU) as the training progresses in order to compare the various difficulty metrics on both datasets (See Figure 3).

6 Results & Analysis

On Table 3, we observe that *soft edit distance* (SED) yields the best performance, outperforming a model that does not use curriculum learning by as much as 2.42 BLEU. It also outperforms all other metrics by roughly 1 BLEU. In general, we see that models perform better on *joint* and *text* than on *data*. This correlates to how a difficulty function is related to the *average bin sizes* of scores it generates. We see that for models that distinguish samples in a more defined manner, it will have a smaller average bin size where probability of having more difficult samples at every confidence threshold is lower. From this, we see that *length* and *DLD* have larger average bin sizes across its difficulty scores and this makes samples less distinguishable from one another. Thus, they result in the smallest improvement over *plain*. We show reordered samples in Table 1 for all difficulty metrics computed *jointly* on data and text. We include length (**L**), rarity (**R**), Damerau-Levenshtein Distance (**DLD**), and the proposed soft edit distance (**SED**).

On the other hand, we also justify the use of weighting for edit operation where *PED*, which is the "hard" variant of *SED* that does not weight edit operations like *SED*, is shown to be far inferior to that of *SED*. The score margin comes up to 2.81 BLEU. Moreover, we further examine the difference in sample orders and observe that *SED* yields more intuitive and better sample ordering as opposed to other metrics.

Human Evaluation. For *human evaluation*, three annotators are instructed to evaluate 100 samples from the *joint* variant to see (1) if the text is *fluent* (score 0-5 with 5 being fully fluent), (2) if it *misses* information contained in the source data and (3) if it includes *wrong* information. These scores are averaged and presented in Table 2.

Metric	Score	Data & Text
L	7	**[Data]** name[Wildwood], eatType[restaurant], familyFriendly[yes] **[Text]** Wildwood restaurant is kid-friendly.
	43	**[Data]** name[Cotto], eatType[coffee shop], food[Indian], priceRange[more than £30], customer rating[high], area[riverside], near[The Portland Arms] **[Text]** Cotto is a coffee shop that offers delicious Indian food, although the price range is high, you will be pleased to know Cotto has high customer ratings. It is located in Riverside near The Portland Arms.
R	17.40	**[Data]** name[Wildwood], eatType[restaurant], familyFriendly[yes] **[Text]** Wildwood restaurant is kid-friendly.
	173.73	**[Data]** name[Cotto], eatType[coffee shop], food[Indian], priceRange[more than £30], customer rating[high], area[riverside], near[The Portland Arms] **[Text]** Cotto is a coffee shop that offers delicious Indian food, although the price range is high, you will be pleased to know Cotto has high customer ratings. It is located in Riverside near The Portland Arms.
DLD	3	**[Data]** name[Wildwood], eatType[restaurant], familyFriendly[yes] **[Text]** Wildwood restaurant is kid-friendly.
	28	**[Data]** name[Cotto], eatType[coffee shop], food[Indian], priceRange[more than £30], customer rating[high], area[riverside], near[The Portland Arms] **[Text]** Cotto is a coffee shop that offers delicious Indian food, although the price range is high, you will be pleased to know Cotto has high customer ratings. It is located in Riverside near The Portland Arms.
SED	17.20	**[Data]** name[The Punter], food[English], priceRange[high] **[Text]** The Punter is a restaurant with high prices.
	87.68	**[Data]** name[The Wrestlers], eatType[coffee shop], food[Italian], priceRange[less than £20], area[city centre], familyFriendly[no], near[Raja Indian Cuisine] **[Text]** The Wrestlers is a coffee shop located in the center of the city that is near Raja Indian Cuisine. It is not family-friendly and the price range is less than 20 pounds.

Table 1: Ordered samples with difficulty metrics.

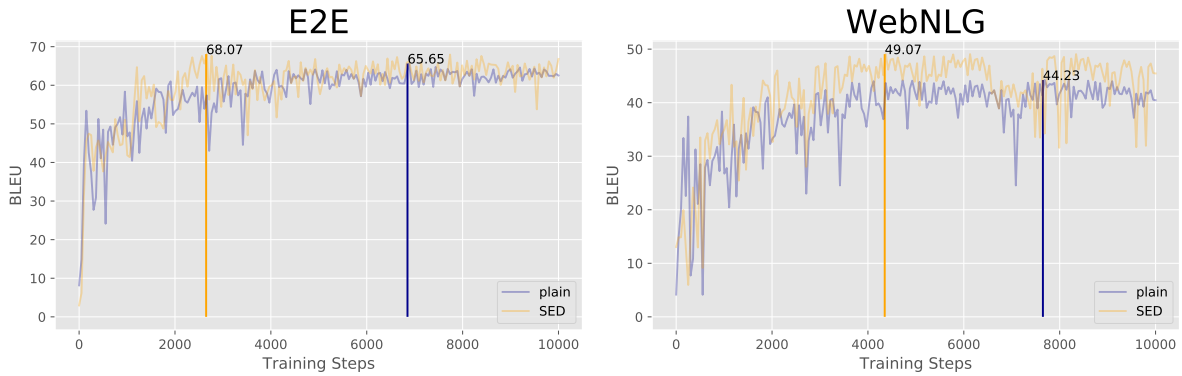


Figure 3: A plot of performance (BLEU) versus the number of steps for E2E and WebNLG datasets. **Vertical bars** indicate where the maximum BLEU scores are attained for **plain** and **SED**.

	source	plain	L	R	DLD	PED	SED
BLEU	data	|	65.34	66.41	-	-	
	text	65.65	65.77	67.01	-	-	-
	joint	|	66.21	67.29	66.58	66.26	**68.07**
Bin Size	data	-	7010.17	385.88	-	-	-
	text	-	737.91	1.04	-	-	-
	joint	-	25.0	1.04	32.26	21.74	**1.02**
Human	fluency	4.35	4.28	4.32	**4.60**	4.23	4.54
	miss	22	16	15	11	14	**9**
	wrong	9	5	7	10	6	**4**

Table 2: Ablation studies for the impact of difficulty metrics on data, text or both (joint) with normalized scores including length (**L**), rarity (**R**), Damerau-Levenshtein Distance (**DLD**), plain edit distance (**PED**), and the proposed soft edit distance (**SED**). **Plain** means no curriculum learning techniques are added. All scores are computed based on the E2E corpus, consisting of both performance (BLEU) and the **average bin size**. Each *bin* is defined by the number of training samples with the same difficulty scores.

On Training Speed. We define speed by the number of updates it takes to reach a performance plateau. On Figure 3, the speedup is measured by the difference between the vertical bars. It can be observed that curriculum learning reduces the training steps to converge, where it consists of up to 38.7% of the total updates for the same model without curriculum learning (on E2E). Further, we see that the use of curriculum learning yields slightly worse performance in the initial training steps, but rise to a higher score and flattens as it converges.

7 Conclusion

To conclude, we show that the sample order does indeed matter when taking into account model competence during training. Further, we demonstrate that the proposed metrics are effective in speeding up model convergence. Given that curriculum learning can be combined with pretty much any neural architecture, we recommend the use of curriculum learning for data-to-text generation. We believe this work offers insights into the annotation process of data with text labels where reduced number of labels are needed.

Acknowledgements

This research was funded in part by the German Research Foundation (DFG) as part of SFB 248

"Foundations of Perspicuous Software Systems". We sincerely thank the anonymous reviewers for their insightful comments that helped us to improve this paper.

References

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 141–148, Ann Arbor, Michigan. Association for Computational Linguistics.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Eric Brill and Robert C Moore. 2000a. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th annual meeting of the association for computational linguistics*, pages 286–293.

Eric Brill and Robert C. Moore. 2000b. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 286–293, Hong Kong. Association for Computational Linguistics.

Emilie Colin, Claire Gardent, Yassine M'rabet, Shashi Narayan, and Laura Perez-Beltrachini. 2016. The WebNLG challenge: Generating text from DBPedia data. In *Proceedings of the 9th International Natural Language Generation conference*, pages 163–167, Edinburgh, UK. Association for Computational Linguistics.

Fred J Damerau. 1964. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176.

Jeffrey L Elman. 1993. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems*, pages 11181–11191.

Gholam Reza Haffari. 2009. *Machine learning approaches for dealing with limited bilingual data in statistical machine translation*. Ph.D. thesis, School of Computing Science-Simon Fraser University.

Tom Kocmi and Ondřej Bojar. 2017. Curriculum learning and minibatch bucketing in neural machine translation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 379–386.

Kai A Krueger and Peter Dayan. 2009. Flexible shaping: How learning in small steps helps. *Cognition*, 110(3):380–394.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213.

Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

Xuebo Liu, Houtim Lai, Derek F Wong, and Lidia S Chao. 2020. Norm-based curriculum learning for neural machine translation. *arXiv preprint arXiv:2006.02014*.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206.

Ankur P Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. Totto: A controlled table-to-text generation dataset. *arXiv preprint arXiv:2004.14373*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom M. Mitchell. 2019. Competence-based curriculum learning for neural machine translation. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1:1162–1172.

Kim Plunkett and Virginia Marchman. 1993. From rote learning to system building: Acquiring verb morphology in children and connectionist nets. *Cognition*, 48(1):21–69.

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263.

Xuan Zhang, Gaurav Kumar, Huda Khayrallah, Kenton Murray, Jeremy Gwinnup, Marianna J Martindale, Paul McNamee, Kevin Duh, and Marine Carpuat.

2018. An empirical exploration of curriculum learning for neural machine translation. *arXiv preprint arXiv:1811.00739*.

Xuan Zhang, Pamela Shapiro, Gaurav Kumar, Paul McNamee, Marine Carpuat, and Kevin Duh. 2019. Curriculum learning for domain adaptation in neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1903–1915.

Yikai Zhou, Baosong Yang, Derek F Wong, Yu Wan, and Lidia S Chao. 2020. Uncertainty-aware curriculum learning for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6934–6944.