Do Data-based Curricula Work?

Maxim K. Surkov¹, Vladislav D. Mosin¹, Ivan P. Yamshchikov¹

¹LEYA Lab, Yandex, Higher School of Economics

Abstract

Current state-of-the-art NLP systems use large neural networks that require lots of computational resources for training. Inspired by human knowledge acquisition, researchers have proposed curriculum learning, — sequencing of tasks (task-based curricula) or ordering and sampling of the datasets (data-based curricula) that facilitate training. This work investigates the benefits of data-based curriculum learning for large modern language models such as BERT and T5. We experiment with various curricula based on a range of complexity measures and different sampling strategies. Extensive experiments on different NLP tasks show that curricula based on various complexity measures rarely has any benefits while random sampling performs either as well or better than curricula.

Introduction

In the last years state-of-art results in natural language processing (NLP) are often obtained with Transformer-like architectures based on the self-attention mechanism (Vaswani et al. 2017) such as BERT (Devlin et al. 2019), GPT-3 (Brown et al. 2020), T5 (Raffel et al. 2020), which could have billions of parameters. Due to the large number of parameters, these architectures require lots of time and hardware resources to be trained.

Curriculum learning (CL) is one of the popular methods to reduce training time and increase the resulting quality of the model. Inspired by the importance of properly ordering information when teaching humans (Avrahami et al. 1997), curriculum learning increases the difficulty of training samples shown to the model over time (Elman 1993). Previous studies have demonstrated that curriculum learning greatly impacts training time and quality in different machine learning domains, such as computer vision (So-viany 2020; Wang et al. 2018) and reinforcement learning (Narvekar et al. 2020; Narvekar 2017). In NLP some results hint that CL might be beneficial (Platanios et al. 2019; Xu et al. 2020; Kocmi and Bojar 2017), however, these results are not as optimistic as in reinforcement learning setup. One of the possible explanations for that is that we still have no viable measure of semantic information for natural languages (Lin and Tegmark 2017).

In contrast to curriculum learning, some papers propose anti-curriculum learning, which kindly reverses training procedure - starts from more difficult samples and continues with easier ones. Counter-intuitively, anti-curriculum learning can be as good as or better than curriculum learning in certain scenarios (Kocmi and Bojar 2017; Zhang et al. 2018). Wu, Dyer, and Neyshabur (2021) notice the conflict between curriculum and anti-curriculum and perform numerous experiments in computer vision, which show that curriculum, as well as anti-curriculum, perform the same or even worse than random sampling on standard tasks. However, curriculum, not anti-curriculum, shows great results on synthetic tasks: learning on noisy datasets and with a little amount of data.

Moreover, recent research in curriculum learning may be divided into two main categories: task-driven curriculum and data-driven curriculum. The idea of the task-driven curriculum was inspired by human behavior. First the model learns how to solve a simple task and then the difficulty is gradually increased. This type of curriculum proposed by Bengio et al. (2009) is considered to be the classical and majority of curriculum-related results are obtained in this framework. Alternatively to the task-driven curriculum, data-driven curricula try to use some form of filtering or sorting of training data that could potentially facilitate learning of a model on a given task.

This paper is an attempt to understand when data-driven curriculum learning works for transformer-based language models. Generally, data-driven curriculum learning may be divided into two main parts: somehow estimating the complexity of the dataset and designing a particular sampling strategy for a given dataset. In the first part of the paper, we list complexity measures applicable to natural language processing. In the second part, we discuss possible sampling strategies that might be applicable for corresponding complexity measures. We run extensive experiments with different metrics and sampling strategies on three different classes of NLP tasks: unsupervised learning with masked language modeling, text classification, and machine translation. Though Platanios et al. (2019) researches data-driven curriculum learning for neural machine translation and shows that even a relatively straight-forward curriculum...
improves the quality of translation, we are not aware of previous works, investigating the impact of curriculum learning for transformer-based architectures applied to MLM or classification. Our experiments show that on all metric-sampling strategy setups data-driven curriculum learning does not give quality increase or time reduction, and often makes results even worse. However, if training data is corrupted with noise, curriculum learning reduces the training time of the model.

Metrics
The first important part of the curriculum learning pipeline is measuring the complexity of samples for a given dataset. Texts could have a complex structure and their complexity can be measured in different ways. A variety of heuristically motivated methods is accompanied by several metrics based on certain aspects of information theory. Let us briefly review them here.

Heuristic Approaches to Text Complexity
The first idea is to determine the complexity of the text as its length. Despite its simplicity, this method is used in different works (Platanios et al. 2019; Kocmi and Bojar 2017). The next family of approaches boils down to phonological, morphological, lexical, or syntactic metrics, derived with some form of expert linguistic knowledge. These methods show controversial results. On one hand, Kurdi (2020) shows promise in terms of classification by difficulty, on another, van der Sluis and van den Broek (2010) used Wikipedia and Simple Wikipedia corpora to demonstrate that language-based metrics do not correlate with the common sense text complexity. The third class of methods treats text as a bag of words and builds metrics based on the frequency analysis. For example, every word can get a rank based on its position in the dictionary sorted by the number of word appearances in a corpus. In this case, complexity may be measured as a maximum rank among the words in a bag (Kocmi and Bojar 2017), this metric is called max frequency rank. Another possible metric is called likelihood. The metric calculates the probability of the text under the assumption that all other possible metric is called likelihood. The metric calculates the probability of the text under the assumption that all tokens are independent, just by multiplying probabilities of all tokens in the text (Platanios et al. 2019). Another metric from this group is TF-IDF (Aizawa 2003), which is widely used in search systems. Finally, the last array of methods is based on using different neural network losses as a complexity measure of a sample.

Using Information Theory for Text Complexity
In this paper, we explore the metrics initially proposed by Ay et al. (2006) to measure the complexity of finite systems and try to see if these metrics could be applied to NLP tasks. Ay et al. (2006) observes that for finite systems not only a set of parts impacts the complexity of the system but also inter-dependencies of the parts. In the context of NLP, this means that text is more than just a bag of words. Authors propose four different metrics to estimate the complexity of a system, however, one of these metrics maximizes on single-letter texts, such as “Aaaaaaaaa”, while the second was created to measure cyclic sequences and does not apply to texts. Let us look at two metrics, namely, TSE and excess entropy, and adapt them to the complexity of texts.

Let \( X_V = (X_{v1}, X_{v2}, \ldots) \) be a sequence of random variables from set \( V = (v1, v2, \ldots) \), and \( A \) is a subset of \( V \), then \( X_A \) is a subsequence of \( X_V \) with elements from \( A \). Let’s determine \( H(X_A) \) as entropy of sequence \( X_A \). However, texts consist of words or tokens, which are not random variables. We propose the following procedure of transforming texts into random variable sequences. For each token in position \( i \) we compute the percentage of texts with this token on the same position and replace the original token with binary distribution with a probability of one equal to the calculated percentage. After transforming text into a sequence of such random variables we can compute its entropy.

\[
H(X_V) = H(X_{v1}) + H(X_{v2}|X_{v1}) + H(X_{v3}|X_{v2}, X_{v1}) + \ldots
\]

If one wants to apply this formula one has to compute entropy for many different conditional distributions while these distributions depend on the order of tokens in a text. First, direct application of the formula would overfit a specific text since all texts are different in a corpus. Second, such computation could not be carried out in a reasonable time. The limit context for conditional distributions to the nearest neighbors one obtains the following formula

\[
H(X_V) = H(X_{v1}) + \sum_{i=2}^{#V} H(X_{v_i}|X_{v_{i-1}})
\]

Using this approximation for entropy one can compute excess entropy (EE) and the complexity measure Tononi, Sporns and Edelman (TSE), (Tononi, Sporns, and Edelman 1994) as they are formulated by Ay et al. (2006)

\[
EE(X_V) = \left[ \sum_{n \in V} H(X_{V_{n:n}}) \right] - (n-1)H(X_V), \quad (1)
\]

\[
TSE(X_V) = \sum_{k=1}^{n-1} \frac{k}{n} C(k)(X_V), \quad (2)
\]

where \( n \) is a size of set \( V \) and

\[
C(k)(X_V) = \sum_{A \subseteq V, |A| = k} H(X_A) - H(X_V).
\]

Samplers
The second important part of curriculum learning is the sampling strategy (or sampler) - the algorithm deciding which samples should be shown to the model at which moment. Let us observe existing curricula and suggest some new ones.

Competence-based. CB
A competence-based curriculum, offered by Platanios et al. (2019), is a classic curriculum, which uniformly samples from increasing dataset’s prefix. Competence is a simple function \( c(t) \), which defines the size of the dataset prefix.

\[
c(t) = \min \left( 1, \sqrt{\frac{1 - c_0^2}{T} + c_0^2} \right)
\]
Where $T$ - total number of steps, $t$ - current step, $c_0$ - hyperparameter set to 0.01.

**Hyperbolic. HYP**
The main idea of this sampler is to increase average batch complexity through time. All samples are split by complexity into $N$ sequential buckets with equal size. Training time is divided into $N$ epochs and the probability of sampling the element from the $j$-th bucket on the $i$-th epoch is proportional to the distance between $j$ and $i$.

$$Pr_r(j) = \frac{c}{|j - i|^{0.5}}$$

Where $Pr_r(j)$ - probability to sample from $j$-th bucket on the $i$-th epoch, $c$ - constant to guarantee that sum of all probabilities equals to 1.

**Difficulty-based. DB**
This sampler is a reversed version of the competence-based one. Instead of sampling from gradually increasing prefix, a difficulty-based sampler takes elements from linearly decreasing suffix.

**Sort-shuffle. SS**
All previously described samplers do not guarantee that each element in the training data would be seen by the model. Sort-shuffle samples each element exactly once by randomly splitting into batches and sorting batches by average complexity.

**Sort-merge. SM**
Many complexity estimates correlate with the length of the text. The main idea of a sort-merge sampler is to remove this correlation and train the model on stable length distribution. This algorithm consists of four main steps:

1. sort dataset by length;
2. sequentially split into buckets;
3. sort each bucket by a complexity metric;
4. form $i$-th batch from $i$-th elements from each bucket.

The sort-merge sampler, like a sequential one, shows each element to the model exactly once.

Equipped with the list of metrics and curriculum samplers we can move on to experimental results.

**Experiments**

We carry our experiments on three different NLP tasks: masked language modeling, text classification, and machine translation. All the experiments are performed with the HuggingFace library (Wolf et al. 2020). Using an inheritance mechanism one could implement different sampling strategies without any changes in the source code of the model. HuggingFace provides the models with their setups, such as hyperparameters and tokenizers. In our experiment we did not change default parameters, so every experiment could be specified with the dataset and the model. We use the base version of the BERT model (Devlin et al. 2019) for MLM and classification and the small version of the T5 model (Raffel et al. 2020) for machine translation. Experiments were performed on BooksCorpus dataset for MLM, Sentiment140 and Hyperpartisan News Detection for classification, and WMT16-en-de for machine translation. To estimate the convergence speed of the curriculum we calculate the average number of steps to reach a specific threshold predefined for every problem.

**Pretraining MLM**

Figure 1 shows the results of MLM pretraining of BERT on BooksCorpus. Irrespective of sampling the complexity measures have similar ranking in terms of their performance on MLM: length, likelihood, TSE, EE, TF-IDF, maximum word rank. Since sorted sampler takes length into account by design, it is not included in the corresponding plots.

On the presented chart a significant slowdown in model convergence speed can be seen for all curriculum learning setups in comparison with the baseline (standard random sampling) learning regime. One can also divide all metrics into two distinct groups. The first one consists of maximum word rank and TF-IDF. The second group includes EE, TSE, likelihood, and length. The metrics in the first group allow the model to converge to a lower loss value. However, the metrics of the second group not only hinder the convergence but also seem to have higher saturation loss. Hence, it is really difficult to find a universal threshold to reasonably compare all metrics and sampler. One should also note that only maximum word rank does not degrade the model quality in comparison with the baseline while curricula other metrics cause severe deterioration. Finally, the last main observation is that curriculum learning, unfortunately, does not allow us to run MLM faster. Moreover, the number of training steps needed to reach a given threshold could be several times higher in comparison with the baseline approach. Table 1 illustrates this fact.

**Text Classification**

Table 2, Table 3 and Figure 3 summarize the experiments with BERT for text classification. Setting an accuracy threshold of 85.5% (98.2% of final model accuracy) on the Sentiment140 dataset one could see that there is a significant difference in convergence for curricula that use different metrics. TF-IDF, TSE, and maximum word rank lead to speedup of 3% on average. Other metrics slow down the model convergence speed. Also, when increasing the dataset size (sentiment140 → HND) the set of effective metrics $\{\text{TF-IDF, maximum word rank, TSE}\}$ is further extended with other metrics. This effect is consistent with the experiments on the pretraining task, where the most effective metrics were maximum word rank and TF-IDF. Length is the worse metric to organize curriculum in all experiment configurations. The one more important conclusion is that the model not always can estimate the complexity of the sample according to the internal state (MLM-loss does not speed

---

1. We publish our code [https://github.com/maximumSHOT-HSE/CurriculumLearning](https://github.com/maximumSHOT-HSE/CurriculumLearning)
2. Experiments were run with Kostenetskiy, Chulkevich, and Kozyrev (2021) hardware
3. [https://huggingface.co/datasets/bookcorpus](https://huggingface.co/datasets/bookcorpus)
4. [https://www.kaggle.com/kazanova/sentiment140](https://www.kaggle.com/kazanova/sentiment140)
5. [https://huggingface.co/datasets/hyperpartisan_news_detection](https://huggingface.co/datasets/hyperpartisan_news_detection)
6. [https://huggingface.co/datasets/wmt16](https://huggingface.co/datasets/wmt16)
Figure 1: Loss function dependency on the number of training steps on MLM for BooksCorpus dataset during the first 40k steps of training. Every plot depicts results for six different complexity estimates combined with a specific sampler.

Figure 2: Various samplers used with TSE complexity measure on WMT16 dataset. Baseline random sampling either outperforms or goes in line with curricula.

Neural Machine Translation

Figure 2 and Table 4 show the experiments with T5 model for machine translation and various curricula. We use the BLEU metric to estimate the quality of the resulting models, however, due to the high standard deviation of this metric we replace the speed estimation with quality estimation. We calculate the average BLEU score over 10 validations from 50k to 100k steps. Once again curriculum learning does not give any benefits for machine translation task.

Under standard conditions, curriculum learning is not able to increase quality or reduce training time for transformer-based models. On MLM tasks all curricula underperform in comparison with random training. One of the possible explanations for this is that longer context could be helpful for the reconstruction of a missing word, yet most complexity measures have a positive correlation with length. On clas-
Table 1: The average number of steps needed to reach given threshold for all configurations metric-sampler on pretraining on BooksCorpus dataset. Maximal deviation for 3 runs is less than $3k$ steps.

| Metrics    | Threshold | Samplers | min loss |
|------------|-----------|----------|----------|
|            |           | CB       | DB       | Hyp      | SS       | SM       |
| max wr     | 2.00      | $\infty$ | 17.5k    | 16.5k    | 16.5k    | 27k      | 1.58     |
| TF-IDF     | 2.00      | $\infty$ | 34k      | 35k      | 37.5k    | $\infty$ | 1.84     |
| EE         | 3.50      | $\infty$ | 4k       | 3.5k     | 4.5k     | 9.5k     | 2.25     |
| TSE        | 3.50      | $\infty$ | 9k       | 9k       | 8.5k     | 18k      | 2.60     |
| likelihood | 3.50      | $\infty$ | 13.5k    | 13.5k    | 15.5k    | 50k      | 2.83     |
| length     | 3.50      | $\infty$ | 50.5k    | $\infty$ | -        | -        | 3.45     |
| baseline   | 2.00      | $\infty$ | 9.5k     |          |          |          | 1.58     |

Table 2: The average number of steps needed to reach given threshold for all configurations metric-sampler on text classification task on sentiment140 dataset. Maximal deviation for 3 runs is less than $3k$ steps.

| Metrics    | Threshold | Samplers | Accuracy |
|------------|-----------|----------|----------|
|            |           | CB       | DB       | Hyp      | SS       | SM       |
| length     | 85.5%     | 112.5k   | 20k      | 19k      | -        | -        | 86.2%    |
| TF-IDF     | 85.5%     | 115.5k   | 21.5k    | 19.5k    | 16.5k    | 22k      | 86.7%    |
| TSE        | 85.5%     | 95.5k    | 16.5k    | 20.5k    | 21.5k    | 18k      | 86.8%    |
| EE         | 85.5%     | 59k      | 19.3k    | 23k      | 20k      | 19k      | 86.7%    |
| max wr     | 85.5%     | 70k      | 18.5k    | 19.5k    | 17k      | 19k      | 86.7%    |
| likelihood | 85.5%     | 112k     | 17.5k    | 21.5k    | 17.5k    | 21.5k    | 86.7%    |
| MLM-loss   | 85.5%     | 59.5k    | 21k      | 23.5k    | 19.5k    | 20k      | 86.1%    |
| baseline   | 85.5%     | 17.5k    |          |          |          |          | 87%      |

sification and machine translation tasks difference with random baseline is not that drastic. Some curricula setups even show a slight improvement in terms of model convergence. Though Platanios et al. (2019) reports competence-based sampling to be beneficial for machine translation we could not reproduce this result in transformer-based architectures, neither using code posted on github, neither code written by us. It seems that while some curricula might be useful for smaller architectures they have no benefits for larger ones.

**Data-based Curricula for Other Architectures**

Since we have received negative results with attention-based architectures we conduct additional experiments with RNNs: two bidirectional LSTMs, dropout, and dense layers. These experiments could illustrate if the negative result obtained above is associated with certain properties of attention-based architectures or could be reproduced with various artificial neural networks.

We run our experiments on Sentiment 140 with 90% train and 10% test split. The curricula include Hyperbole, Difficulty-Based and Competence-Based samplers, and TSE and length difficulty metrics. Figure 4 shows that a data-driven curriculum does not have a significant influence on the results.

**Discussion**

After a series of extensive experiments with different NLP tasks, various samplers, complexity measures, and transformer-based models we could not see any significant and persistent quality increase or training time reduction. Wu, Dyer, and Neyshabur (2021) report similarly discouraging results for data-driven curriculum learning in computer vision. The only case when data-driven curriculum learning was shown to be useful in machine vision was dealing with images that had synthetically inserted noise. Let us see if similar results could be valid for NLP. To do that we experiment with three different noises:

- keyboard noise (Srivastava, Makhija, and Gupta 2020) - randomly change some letters into the near ones on the keyboard;
- random swaps - randomly swap some letters in the words;
- spelling errors - emulating mistakes which real people usually do while writing texts.

To add noise to the text we choose $p \sim U[0, p_{max}]$ - the percentage of changed symbols from uniform distribution and apply noise operation to such part of symbols. With these types of noise, one could have a natural complexity measure for noisy data. The average amount of tokens per word (or TPW) naturally corresponds with the amount of noise introduced into the text. Indeed, as we randomly change the letter, its combinations with neighbors become unfrequent and the tokenization procedure splits the word into different tokens. One can easily carry out a range of experiments to see that the TPW metric is highly correlated with the amount of introduced noise.

Table 5 demonstrates the results of experiments with Sentiment140 corrupted with the noise as described above. Experiments showed that the behavior is similar across all three
Table 3: The average number of steps needed to reach given threshold for all configurations metric-sampler on text classification task on Hyperpartisan News Detections dataset. Maximal deviation for 3 runs is less than 3k steps.

| Metrics      | Threshold | Samplers | Accuracy |
|--------------|-----------|----------|----------|
| length       | 92.9%     | CB 55k DB 23k Hyp 22.5k SS - SM - | 93.7% |
| TF-IDF       | 92.9%     | ∞ 19.5k | 24k 23.5k SS 33k | 93.5% |
| TSE          | 92.9%     | 56.5k 21k | 23k 22k SS 31k | 93.8% |
| EE           | 92.9%     | 71.5k 25.5k | 22.5k 19.5k SS 32.5k | 93.8% |
| max wr       | 92.9%     | ∞ 22k | 20.5k 22.5k SS 39k | 93.6% |
| likelihood   | 92.9%     | ∞ 20k | 24k 20k SS 30k | 93.8% |
| MLM-loss     | 92.9%     | 23.5k 18k | 23k 24k SS 20k | 93.9% |
| baseline     | 92.9%     | 22k     |         | 93.8% |

(a) Sentiment140 with sort-merge sampler for all complexity measures.
(b) Sentiment140 with max word rank complexity measure for all samplers.
(c) Hyperpartisan News with sort-shuffle samples for all complexity measures.
(d) Hyperpartisan News with max word rank complexity measure for all samplers.

Figure 3: Pre-trained BERT fine-tuned on Sentiment140 and Hyperpartisan News Detection datasets. Accuracy of the classifier as a function of the number of training steps.

types of noise so we only discuss the noise of the keyboard further.

With a higher noise level \( p = 0.4 \) one could see that 83.5% accuracy threshold could be reached with 2.5k steps with curriculum learning instead of 4.3k steps with the standard randomized algorithm. As the threshold increases, the profit becomes smaller and smaller. This stands to reason since the curriculum learning model is trained on the clean examples at first and smoothly moves on to examples with more noise.

Curriculum learning firstly appeared in reinforcement learning and was used to increase quality, the main idea was to train an agent on a simple task until it will be able to solve it and only after that increase the difficulty of the
(a) Sentiment140 with length as complexity metric and three samplers.

(b) Sentiment140 with TSE as complexity metric and three samplers.

Figure 4: Test results with LSTM on Sentiment140 dataset. Accuracy of the classifier as a function of the number of training steps.

Table 4: The average BLEU score from 50k to 100k steps on WMT16 dataset

| Metrics | Samplers | CB | DB | Hyp | SS | SM |
|---------|----------|----|----|-----|----|----|
| length  |          | 10.1 | 17.4 | 16.3 | -  | -  |
| TSE     |          | 10.3 | 18.4 | 16.8 | 13.8 | 14.8 |
| EE      |          | 10.2 | 18.2 | 16.9 | 13.3 | 15.0 |
| baseline|          |      |     |     |      | 18.3 |

Table 5: The average number of steps needed to reach given threshold accuracy on text classification task with competence-based sampler on sentiment140 dataset and noise. 40% of letters in sentences are corrupted with noise. TPW based curriculum is beneficial for the convergence of the model. The maximal deviation for 3 runs is less than 1k steps.

| Metrics | Threshold | Steps |
|---------|-----------|-------|
| TSE     | 83.5%     | 5k    |
| TPW     | 83.5%     | 2.5k  |
| baseline| 83.5%     | 4.3k  |

In this work, we ran extensive experiments with curriculum learning for transformer-based architectures on three NLP tasks: masked language modeling, text classification, and machine translation. We demonstrate that curricula do not help in the standard training setting and sometimes even make results worse. We also demonstrate that curriculum might be useful on the tasks where data is significantly corrupted with external noise. In such cases, a curriculum may nearly double the performance of the model in the earlier stages.

References

Aizawa, A. 2003. An information-theoretic perspective of tf–idf measures. Information Processing & Management, 39(1): 45–65.

Avrahami, J.; Kareev, Y.; Bogot, Y.; Caspi, R.; Dunaevsky, S.; and Lerner, S. 1997. Teaching by examples: Implications for the process of category acquisition. The Quarterly Journal of Experimental Psychology Section A, 50(3): 586–606.

Ay, N.; Olbrich, E.; Bertschinger, N.; and Jost, J. 2006. A unifying framework for complexity measures of finite systems. In Proceedings of ECCS, volume 6. Citeseer.

Bengio, Y.; Louradour, J.; Collobert, R.; and Weston, J. 2009. Curriculum learning. In Proceedings of the 26th annual international conference on machine learning, 41–48.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M. F.; and Lin, H., eds., Advances in Neural Information Processing Systems, volume 33, 1877–1901. Curran Associates, Inc.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics.
Elman, J. L. 1993. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1): 71–99.

Kocmi, T.; and Bojar, O. 2017. Curriculum Learning and Minibatch Bucketing in Neural Machine Translation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, RANLP 2017, 379–386.

Kostenetskiy, P.; Chulkevich, R.; and Kozyrev, V. 2021. HPC Resources of the Higher School of Economics. In *Journal of Physics: Conference Series*, volume 1740, 012050. IOP Publishing.

Kurdi, M. Z. 2020. Text Complexity Classification Based on Linguistic Information: Application to Intelligent Tutoring of ESL. *Journal of Data Mining & Digital Humanities*, 2020.

Lin, H. W.; and Tegmark, M. 2017. Critical behavior in physics and probabilistic formal languages. *Entropy*, 19(7): 299.

Narvekar, S. 2017. Curriculum Learning in Reinforcement Learning. In *IJCAI*, 5195–5196.

Narvekar, S.; Peng, B.; Leonetti, M.; Sinapov, J.; Taylor, M. E.; and Stone, P. 2020. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21(181): 1–50.

Platanios, E. A.; Stretcu, O.; Neubig, G.; Poczós, B.; and Mitchell, T. 2019. Competence-based Curriculum Learning for 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)*, 1162–1172.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21: 1–67.

Soviany, P. 2020. Curriculum Learning with Diversity for Supervised Computer Vision Tasks. In *MRC@ECAI*.

Srivastava, A.; Makhija, P.; and Gupta, A. 2020. Noisy Text Data: Achilles’ Heel of BERT. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, 16–21.

Tononi, G.; Sporns, O.; and Edelman, G. M. 1994. A measure for brain complexity: relating functional segregation and integration in the nervous system. *Proceedings of the National Academy of Sciences*, 91(11): 5033–5037.

van der Sluis, F.; and van den Broek, E. L. 2010. Using complexity measures in information retrieval. In *Proceedings of the third symposium on information interaction in context*, 383–388.

Wang, Y.; Perazzi, F.; McWilliams, B.; Sorkine-Hornung, A.; Sorkine-Hornung, O.; and Schroers, C. 2018. A fully progressive approach to single-image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 864–873.

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; Davison, J.; Shleifer, S.; von Platen, P.; Ma, C.; Jernite, Y.; Plu, J.; Xu, C.; Scao, T. L.; Gugger, S.; Drame, M.; Lhoest, Q.; and Rush, A. M. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 38–45. Online: Association for Computational Linguistics.

Wu, X.; Dyer, E.; and Neyshabur, B. 2021. When Do Curricula Work? In *International Conference on Learning Representations*.

Xu, B.; Zhang, L.; Mao, Z.; Wang, Q.; Xie, H.; and Zhang, Y. 2020. Curriculum learning for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 6095–6104.

Xu, B.; Zhang, L.; Mao, Z.; Wang, Q.; Xie, H.; and Zhang, Y. 2020. Curriculum learning for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 6095–6104.

Zhang, X.; Kumar, G.; Khayrallah, H.; Murray, K.; Gwinnup, J.; Martindale, M. J.; McNamee, P.; Duh, K.; and Carpuat, M. 2018. An empirical exploration of curriculum learning for neural machine translation. *arXiv preprint arXiv:1811.00739*. 

Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.