Curriculum Effects in Transformer Language Models with Limited Computation Resources

Benjamin Basseri
CS229BR
Harvard University
basseri@cs50.harvard.edu

Abstract

Properly ordering training information is crucial to human education [Avrahmi 97], but has produced unclear results in deep learning [Wu whend o curricula work]. Curriculum Learning (CL) supposes that presenting examples to a model that gradually increase in complexity may achieve better results than random sampling. While some studies have shown positive effects by training on increasing difficulty, other find effects when ordering training by decreasing difficulty[Kocmi and Bojar] and others still find no effect at all. Adding to this uncertainty is the fact that previous studies span a wide variety of domains and scale of data and computing resources.

Supposing that language modeling (LM) may be more primed to benefit from CL than other domains, this study conducted experiments to compare curriculum effects while holding architecture, corpus, and training time constant. Where CL has been previously observed to be effective, this paper proposes a simpler, more intuitive method to partition training samples by difficulty: the reading level of the source text. We also investigate the effects of changing the easy vs. hard sample proportion through a simpler mechanic than has been used in the past. In experiments we find that even in small scale models, the architecture is indeed agnostic to CL, although a small but noticeable curriculum effect appears in syntax learning that may warrant further research.

1 Introduction

1.1 Curriculum Effects

Based on the notion that humans learn better when training data is subject to certain order [], curriculum learning (CL) explores whether similar effects can be found in deep learning. Previous studies have investigated empirical CL effects in diverse domains such as language modeling [], computer vision [], and evolutionary computing []. Some studies have found positive results by ordering training data by increasing difficulty []. Others have found precisely the opposite and propose anti-curricula [] wherein training data is ordered by decreasing difficulty. Yet others find little or no curriculum effect []. As a result of these conflicting results, CL techniques have not found widespread adoption [when does].

Adding to the confusion, these empirical studies have taken place across diverse domains with varying architecture and computing resources. Training data size and computing resources are understood to be the main drivers of model performance [], so it seems plausible that curriculum effects may only be detectable when holding such values constant.

Furthermore, if CL has any effect it may be particular to natural language. NLP endeavors to understand language as it is used *by humans*, Humans do tend to learn language in an ordered, structured fashion and it is possible that ultimately our usage reflects that ordering. In contrast,

computer vision models are often required to learn an internal representation of images whose structure depends less on human input and more on physics and biology. In other words, natural language structure comes from human usage while photograph images have structure dictated by the physical world.

1.2 Pre-training

In recent years the pre-training framework has led to various powerful language models such as ELMo [], BERT [] and ERNIE []. These pre-trained models have proven to transfer successfully to a variety of natural language processing tasks, including inference, entailment, and named entity recognition [universal LM]. This suggests that an adequately pre-trained model learns a 'good' internal representation of language, after which learning specific NLP tasks require far less effort.

To accomplish pre-training and transfer learning, many implementations use a base model for the pre-training then attach various task-specific 'heads' to the pre-trained base. The base is often 8 to 12 layers deep while a head may be as few as one or two layers. These are not to be confused with the self-attention 'heads' in a transformer [], which these models also tend to use in their architecture.

Most commonly, base models use word co-occurrence as a self-supervised task in order to pre-train. Commonly, Masked Language Modeling is used [], where a model is tasked with predicting masked words from phrases. Thus, such models learn an embedding and encoding primarily by learning word likelihoods conditioned on context.

1.3 Fine-tuning

After a model has been pre-trained, the MLM head is detached and a new 'head' can be appended, one or more layers designed for a particular NLP task. The base and head together are then trained on the task. This has the effect of training the head and 'fine-tuning' the base model, whose weights have already been trained to learn an internal language representation to some degree. As the terminology suggests, the presumption is that the pre-training has moved the base model's weights into an optimal neighborhood such that specific NLP tasks can easily be learned on top of the representations already learned by the base model.

1.4 Placement of CL

Studies have proposed using CL to optimize the fine-tuning of NLP tasks

2 Relevant Work

Word2Vec, glove

Elmo, Bert, Ernie

GPT3

MLM for pre-training but is there an order effect

Wu [When does...] proposes that curricula can improve performance in the context of limited training resources or noisy data however this was on image data, not language data

Xu [CL for NLU] makes up its own metric for difficulty, using a Difficulty Evaluation. It is then begins gradually mixing in hard examples

MLM can be much more simply approached

GPT-3 uses a form of curriculum learning in its approach, however CL's contribution to GPT-3's performance is confounded by its scale.

3 Ordering Effects Explored

Unlike previous studies, these experiments more simply resolve the question of text difficulty and more clearly delineate the CL process. Rather than attempt to quantify the difficulty of a text

according to a Difficulty Evalution as in [Xu], this paper proposes to simply use a text's understood reading level. Coarsely, we may use the commonly accepted grade reading level K-12 as a metric. Estimating differences in 'adult' texts are beyond the scope of this paper. Given that a text is written at a certain reading level, any MLM sample drawn from this text is likely to have roughly the same difficulty as any other. This improves over [Xu] in that it does not require a Difficulty Evaluation be performed.

Since text prepared at various levels of language knowledge are readily available, we may simply use reading grade level, K - 12 to 'adult' as a metric In humans, we take great care to properly order training materials and education in general. Students begin to learn on easy material then advance to more difficult material. In language models, we wish to know if any such effects appear in their training, if training on one type of text before another aids or hinders its ability to learn more language structure.

3.1 Corpus

To investigate, we assembled a custom training corpus of texts. The corpus is relatively small by current standards, 11.3 million words. Ordered by increasing difficulty it includes:

1. *Raggedy-Ann Stories* by John Gruelle (kindergarten reading level)
2. *The Secret Garden* by Frances Hodgson Burnett (2nd grade)
3. *Treasure Island* by Robert Louis Stevenson (7th grade)
4. *A Tale of Two Cities* by Charles Dickens (9th grade)
5. Reuters text set from the Brown Corpus [] (adult)
6. Wikipedia, 2500 randomly selected pages (adult)
7. *The Analysis of Mind* by Bertrand Russell (adult)

3.2 Pre-training

Two training runs were executed. In the first run the model trained on the texts in various orders while in the second run the amount of training time the model allocated to each text was also manipulated.

First Run: permutations. We pre-trained five instances on each of the following permutations:

In order: the models trained on the corpus in order of increasing difficulty

Reverse: the models trained on the corpus in order of decreasing difficulty

Random: the models trained on the corpus with a random ordering of the text. Three different randomizations were trained (the random orderings all had similar performance so we present the results from the one random ordering [1, 2, 6, 3, 5, 4, 7]).

Second Run: Normalized training time. In another round of training runs, models were again created using **In order**, **Reverse**, and **Random** text permutations, but with training time normalized across texts. That is, these models trained on each text for the same number of steps, regardless of the text length. This way each text contributes the same number of gradient updates to the base model. This may be thought to reflect grade-school learning, where each year a student generally studies materials only in that year's grade level. Holding the total train time constant, these configurations were produced to see if emphasizing different subsets of the corpus leads to any meaningful effect in downstream tasks.

Second Run: The Exponential configuration. In the normalized training runs a final configuration, **Exponential**, was produced in which the training time devoted to each text decayed exponentially from easy to advanced texts. In this configuration, the model trained on the easiest text for roughly 50% of its allotted train time, the second easiest text for 25%, and so on.

Multiple instances trained. For each configuration, five separate instances were trained. Originally, the intention was to average the five results since there may have been excess variation due to the relatively small size of the training set. However, across each configuration the five pre-trained instances had nearly identical losses. Therefore, one model from each configuration was used for fine-tuning and evaluation.

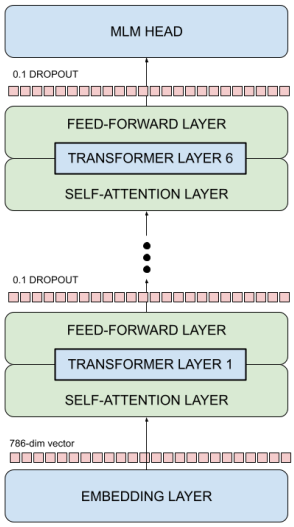

Figure 1: Model architecture used for pre-training, 12 self-attention heads per transformer layer.

3.3 Fine-tuning

Base model + head. The base model from each pre-trained model was then detached from the MLM task. A variety of canonical GLUE tasks [] were selected and for each, a new head was attached to the base model to train on the task. Note that each GLUE task fine-tuned a distinct instance; this means that the base models were re-loaded and attached to new heads for each task. This was done to ensure that fine-tuning on one task would not contribute to a model's performance on another task.

Evaluation metrics. Each was then fine-tuned on a variety of the canonical GLUE tasks and evaluated by the corresponding metric, see tables 1, 2. For CoLA and STSB a correlation metric is used, and accuracy used for all others.

4 Experiment

4.1 Implementation

Corpus. The corpus was constructed to contain a range of texts with clear separability by reading level. The *Reuters* text was extracted from the Brown Corpus via the NLTK Python library []. For the Wikipedia text set, 2500 random articles were downloaded via the Wikipedia Python library []. The remaining texts were sourced from Project Gutenberg [].

Hardware. Models were trained on an Intel Xeon 2.2 GHz server with an NVIDIA Tesla P100 GPU via Google Cloud.

Software. The experiments were programmed primarily using the **transformers** Python library by HuggingFace. The 'fast Roberta tokenizer' was used for text tokenization.

Model architecture. Models were based on the RoBERTa architecture and pre-trained from scratch. This architecture uses a single embedding layer followed by 6 transformer layers and finally an MLM head. We used 12 self-attention heads, an embedded dimension of 768, vocabulary size of 52,000, and maximum sequence length of 512. See figure 1.

Pre-training and fine-Tuning. Each model required approximately 1.5 hours for pre-training and 1 hour for fine-tuning on the above hardware. Each fine-tuning task was trained on a new instance of the saved pre-trained model. That is, each GLUE task was trained independently of each other.

(a) In order model (b) Reverse order model (c) Random order model

Figure 2: Training loss for various order configuration

Table 1: Metrics used for GLUE Scores

CoLA	MRPC
CoLA	Matthews Correlation
MRPC	Accuracy
RTE	Pearson Correlation
STSB	Accuracy
WNLI	Accuracy

4.2 Evaluation

For each model configuration, five separate models were trained. However, since there was only trivial difference between each instance in a given configuration, we report the results from the first instance of each group.

In the first round of training runs we permuted the order in which the model trained on each text. Holding training time constant, the **Reverse** order achieved lower training loss than **In order** and **Random** [1] See figure 2. However, all three configurations of pre-trained models led to almost exactly the same GLUE performance (see table 1).

For the models where training time was normalized across texts, once again there is virtually no difference in performance on the downstream tasks. Of some interest is the fact that the CoLA scores were somewhat higher in these models as opposed to the models which did not normalize training time across texts. We also note that the **Exponential** model achieved strongest performance on CoLA and comparable results on the other tasks.

Code and data can be found at `https://github.com/benjaminb/curriculumeffects`.

5 Conclusion and Future Work

In the first experiment we structured a corpus based on texts of increasing difficulty, then pre-trained several models based on permutations of those texts. After fine-tuning on specific NLP tasks all models achieved nearly identical performance.

In the second experiment we used the same corpus and text permutations but normalized training time across texts, so the model trained an equal number of steps in each text. Effectively, this emphasized

[1] We report results for the random ordering [1, 2, 6, 3, 5, 4, 7].

Table 2: GLUE Scores for Ordered Training

Model	CoLA	MRPC	RTE	STSB	WNLI
In order	0.10	0.71	0.56	0.18	0.56
Reverse	0.11	0.70	0.56	0.19	0.56
Random	0.11	0.70	0.57	0.20	0.53

See Table 1 for score metrics.

Table 3: GLUE Scores for Normalized Training Time, Exponential Emphasis

Model	CoLA	MRPC	RTE	STSB	WNLI
In order	0.15	0.71	0.54	0.22	0.53
Reverse	0.16	0.70	0.54	0.20	0.51
Random	0.15	0.71	0.58	0.21	0.56
Expo	0.19	0.70	0.56	0.23	0.56

See Table 1 for score metrics.

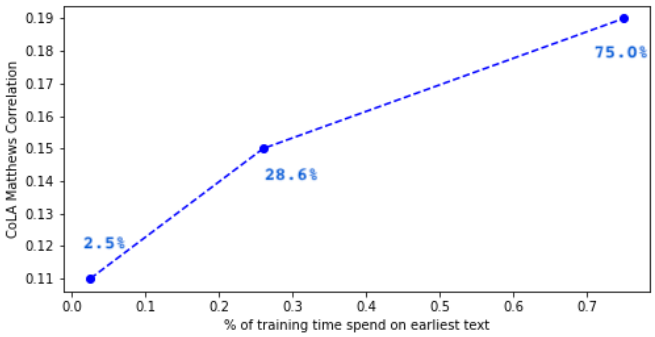

Figure 3: CoLA correlation score by % of training steps in early texts

the lower reading level texts as they tended to be shorter. While this improved CoLA scores overall, once again the models achieved nearly identical performance.

As an addendum to the second experiment, the **Exponential** emphasis allocated much more training time to the lower reading level texts, holding total training time constant. This model achieved comparable results to the others with the notable exception of best CoLA score. This result suggests further research may be worth pursuing in this direction.

Based on the above results, it appears that BERT-type language models are quite agnostic as to training order. The fact that GLUE scores were nearly identical between the normalized training time models and non-normalized ones likewise suggests that the range of content in the corpus may be more important than the relative amount of content per reading level.

The increased CoLA scores in the normalized models does suggest that emphasizing the simpler texts leads to a better understanding of basic grammar, which is essentially the CoLA task. The **Exponential** model's best-in-class performance here adds weight to this hypothesis; in these experiments as more emphasis was given to the earlier texts, the better the model performed on identifying grammatically correct sentences. This may be due to the fact that much of the advanced text came from news and Wikipedia articles, which often contain sentence fragments (headlines, abbreviations, etc.). See figure 3.

5.1 Future work

Scaling Up. The experiments here used a relatively small and focused corpus and small computing resources compared to full-scale language models. It will be interesting to see if these effects continue to hold as corpora size and training time increases, or if there is a threshold after which order effects appear.

Investigating an emphasis effect. Furthermore it may be of interest to see if the corpus emphasis trend seen here continues as models scale up as well. Since a new transformer model must learn how to self-attend, it may be that by first training on basic texts, the training 'grooves in' the basic language structure into the model weights, upon which more advanced text can better be learned.

References

References follow the acknowledgments. Use unnumbered first-level heading for the references. Any choice of citation style is acceptable as long as you are consistent. It is permissible to reduce the font size to `small` (9 point) when listing the references. Note that the Reference section does not count towards the page limit.

[1] Alexander, J.A. & Mozer, M.C. (1995) Template-based algorithms for connectionist rule extraction. In G. Tesauro, D.S. Touretzky and T.K. Leen (eds.), *Advances in Neural Information Processing Systems 7*, pp. 609–616. Cambridge, MA: MIT Press.

[2] Bower, J.M. & Beeman, D. (1995) *The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System.* New York: TELOS/Springer–Verlag.

[3] Hasselmo, M.E., Schnell, E. & Barkai, E. (1995) Dynamics of learning and recall at excitatory recurrent synapses and cholinergic modulation in rat hippocampal region CA3. *Journal of Neuroscience* **15**(7):5249-5262.

[4] Salazar, J., Liang, D., Nguyen, T., Kirchoff, K. Masked Language Model Scoring. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (2020), 2699-2712

[] Francis, W. Nelson & Henry Kucera. 1979. BROWN CORPUS MANUAL: Manual of Information to Accompany a Standard Corpus of Present-Day Edited American English for Use with Digital Computers.

A Appendix