Exploring Curriculum Learning for Natural Language Inference: A Study on Smart Ordering of Training Samples

Tomás Pires Iken
t.m.piresiken@students.uu.nl
Yfke Smit
y.b.m.smit@students.uu.nl
Sarah Abdalla
s.t.m.abdalla@students.uu.nl

Abstract

In the field of artificial intelligence, pre-trained language models like GPT-3 and BERT have become ubiquitous tools for various natural language processing tasks. Fine-tuning these models for specific tasks, such as natural language inference, can be done with different training data, impacting performance results. This paper investigates the concepts of Curriculum Learning, inspired by human learning paradigms, to enhance the performance of pre-trained language models on natural language inference tasks. Specifically, we explore the notion of "smart" ordering of training samples, aiming to reflect increasing levels of difficulty. Using metrics like parsed tree depth and Jaccard similarity between Part-of-Speech tags, we construct various orderings for training data. By training the DistilBERT model on various orderings of the Stanford Natural Language Inference dataset, we evaluate the effectiveness of these smart orderings. Despite various experiments, our findings reveal no improvement in model performance, leaving room for more exploration in Curriculum Learning for natural language inference tasks.

1 Introduction

Not only in the field of artificial intelligence (AI) but also in our daily lives, it is very unlikely that you have never heard of Large Language Models like GPT-3 or Bidirectional Encoder Representations from Transformers (BERT). These pre-trained language models (PLMs) can be fine-tuned for specific tasks, which is very useful and time-saving, for example for companies that want to incorporate AI in their processes.

An example of a task for a PLM is natural language inference (NLI), where the task is to determine if a given hypothesis logically follows from a given premise.

To ensure the performance of these PLMs, several researches have been done in the fine-tuning stage. For example, it has been known that the training example order in Stochastic Gradient Descent (SGD) affects the convergence rate (Lu et al., 2022). Schluter and Varab (2018) found that in the Stanford Natural Language Inference (SNLI) (Bowman et al., 2015) and Multi-Genre Natural Language Inference (MNLI) (Williams et al., 2018) datasets, the performance of a trained neural model is exceedingly sensitive to the order of training samples. Bengio et al. (2009) introduced Curriculum Learning (CL) since they reasoned that when humans learn about a certain idea, they learn better when the examples are organized in a manner that they illustrate more concepts, and are gradually more complex. In CL, these examples symbolize the training samples.

1.1 Research Objective

The idea of using CL in training language models is used as an inspiration for the research in this paper. The purpose of the research in this paper is to find out whether there is a way for the training samples to be ordered in a way such that the order significantly boosts the performance of a fine-tuned model on an NLI task, specifically, we aim to organize the training samples in a way that reflect increasing levels of difficulty, as proposed by Xu et al. (2020). We call these "smart" orderings.

2 Related Work

As stated before, prior research has been done in the field of CL in NLI. In this section, we go more in-depth into existing literature in order to provide a comprehensive overview of the research already conducted in this domain.

Xu et al. (2020) provided valuable insights regarding constructing orders in training examples into their implementation of a CL approach such

that they boost the model's performance. Their work emphasized the significance of organizing training data in an arranged fashion which considers the difficulty levels of examples in natural language understanding (NLU) tasks.

As mentioned in the introduction, Schluter and Varab (2018) found that the performance of a trained neural model is exceedingly sensitive to the order of training samples, more specifically in the SNLI and MNLI datasets. Using statistical learning (to first randomly shuffle the training set) or deep learning engineering strategies (ordering the data by length) for the organization of the training samples has resulted in significantly and substantially lower performance on the mentioned datasets.

The study in Gururangan et al. (2018) found that in a significant portion of the SNLI and MNLI datasets, the hypotheses contain hints which give away the correct classification without needing to see the original premise. This discovery suggests that the SNLI and MNLI datasets might not be reliable, as the achieved accuracy could be higher than it should be due to the bias in the hypotheses.

Ranaldi et al. (2023) built on CL and proposed a novel, linguistically motivated measure, LRC, to determine example complexity for organizing training samples. Their experiments have shown that their additional complexity measure LRC to CL, called CL-LRC, outperforms the existing CL and non-CL methods, more specifically for training BERT and RoBERTa from scratch. Additionally, by analyzing different measures, such as perplexity, loss, and learning curve of different PLMs from scratch, which have shown that their proposed complexity measure CL-LRC outperforms the state-of-the-art.

3 Preliminaries

In this section, some context is given on the dataset and metrics used in order to understand the elements used in the experimental setup.

3.1 Dataset Description

In this paper, we train and evaluate the models using the SNLI dataset (Bowman et al., 2015), where its premises are based on image captions from the Flickr30k corpus (Young et al., 2014). The corresponding hypotheses are generated by crowd-sourced annotators who were shown the premise (Glockner et al., 2018). Consequently, every premise and hypothesis pair is manually labeled with the labels entailment, contradiction, and neutral to receive balanced classification (Bowman et al., 2015).

3.2 Ordering metrics

The model's performance difference is measured by using several metrics to order the SNLI problems in the dataset before training, such as using the depth of a parsed tree, the combination of Part-of-Speech (POS) tags and Jaccard similarity, and the relative height, and using them to construct orders from simplest to most complex of the training examples. The components used in these metrics are explained in the sections below.

Parsed trees. Provided on the SNLI dataset are the parse trees for both the hypothesis and premise sentences. These trees are a representation of the compositional semantics of the sentence and its order of construction, attributing each part of the sentence to a type and connecting them to show how the sentence is constructed. This can be used to determine the complexity of a sentence as a more complex sentence tends to have a larger tree since it has more components.

POS tags. The POS tags are provided by the SNLI dataset. POS tags capture the syntactic information of a given sentence, which can be useful to compare the syntactic complexity between premises and hypotheses.

Jaccard similarity. A useful manner to compare the similarity between two given sentences is the Jaccard similarity. The more common words in between the sentences, the higher the Jaccard similarity. In our research, this is practical when wanting to find out the similarity between the premises and hypothesis.

4 Methodology

In order to find "smart orderings", we created metrics that were used to sort the data before fine-tuning a Language Model.

4.1 Choices related to computational resources

Due to computational scarcity, we made some choices in order to reduce the training time of the models.

4.1.1 Model

In the case of the model, we made use of a distilled version of the BERT model, called DistilBERT

(Sanh et al., 2019). This is a smaller model and therefore has a smaller training time.

4.1.2 Subset Creation

Furthermore, to reduce computational effort, we trained the model on a subset of the SNLI dataset. The subset was created by splitting the SNLI dataset into subsets of 100,000 items and removing the premises from the subset if there were no 3 matching hypotheses in the subset, resulting in subsets varying in size.

These 5 subsets were used as training sets in 4 different orderings: original order, sorted on the premise length, sorted on the hypothesis length, and sorted on the premise + hypothesis length. The results of these subsets were compared with the findings of Schluter and Varab (2018), and the best-performing subset was chosen, which consisted of 99732 training samples.

The data that is used from this subset are the premise, hypothesis, label, ID of the premise (captionID), ID of the combination between premise and hypothesis (pairID), and the trees containing POS tags from both the premise and hypothesis. Before training the data is formatted to a set of premises, hypotheses, and labels. The other items are used for ordering the training set and will not be used during training.

4.2 Metrics for constructing smart orders

We defined several metrics to manipulate the order of the training examples, to observe whether they would lead to a significant boost to the performance of the NLI model compared to the baseline model.

4.2.1 POS tags + Jaccard similarity

One of the metrics consists of the comparison of the POS tags, between the premise and the hypothesis of the NLI problem. For every problem in the training examples, the Jaccard similarity of the POS tags between the premise and hypothesis is measured, which returns a real number. The training examples could be ordered in an ascending or decreasing manner, whichever one is preferred at the given moment, according to the similarity's real number. As the POS tags capture the syntactic information of the premises and the hypotheses, in the decreasing manner we would start with simpler examples where the syntactic structures are more similar and gradually progress toward more complex examples.

Furthermore, the Jaccard similarity between the words of the premise and hypothesis is used as a metric. It is also expected here that a higher similarity score represents an easier training sample.

4.2.2 Depth of parsed tree

Another metric compares the depth of the parsed tree between the premise and the hypothesis for every problem in the training examples. The arrangement of the order of the training examples can be achieved by sorting them based on the ascending or descending absolute numerical difference between the premise and hypothesis depths, depending on what the preference is at the given moment. The depth of a parsed tree presents the sentence's structural complexity. A small absolute difference between the depth of the tree of a premise and the hypothesis indicates similar syntactic structures. Conversely, a larger absolute difference could suggest that the premise and hypothesis have contrasting syntactic structures. For the ascending manner, we would start with examples that have a more similar depth, and so are more similar and simpler for the model, and gradually receive sentences with trees involving more complex contrasting patterns in the sentences.

4.2.3 Relative height

The last metric is the relative depth of the tree. That is the depth of the tree divided by the number of words in the sentence. This is done for both the absolute difference of relative heights in the premise and the hypothesis and the sum of the relative heights of the two. As mentioned in section 4.2.2 a more complex sentence will have a larger tree, however, sentences can be of different lengths and still have the same depth of tree. This metric attempts to take into account shorter sentences with large trees as more complex than large sentences with the same tree which convey the same complexity of information but with more words. Therefore creating a complexity density of sorts. We can then sort the items from those with the least complexity density to those with the highest. Using both the sum and difference between premise and hypothesis of these densities we can determine the total complexity density and contrasting complexity density of each item pair.

4.3 Keeping triplets together

As explained before, the SNLI dataset consists of premise-hypothesis triplets. For every premise,

there are three hypotheses. At the beginning of our research, this was not taken into account in our orderings, resulting in separating these samples in the training set.

However, in the original ordering, every premise-hypothesis triplet is kept together. There is a possibility that this improves training performance in comparison to randomly ordering the dataset and not keeping the triplets together.

Therefore, the ordering functions using the metrics described above were rewritten so that for every premise, the mean value of the metric score from its hypotheses was taken. This mean value is used to sort the samples, keeping the triplets together.

5 Experimental Setup

Several experimental setups have been tried to find a significant effect on performance, for example, training on a subset of 250.000 samples, training with random orderings, and training with different metrics.

Before running the experiments in order to see if the different orderings had a significant effect on performance, the necessary elements were imported and formatted for our use. All training was done using the V100 GPU available on Google Colab.

5.1 Components

As stated before, we used the SNLI dataset. We imported the original one instead of the SNLI dataset present in the Hugging Face library, in order to use the parse trees. Before training, the evaluation and test data is preprocessed and encoded using the tokenization function from the Hugging Face library.

Furthermore, our own resources are imported, like the subset that we created before that is used for training, and the functions used for ordering this subset in different ways.

5.2 Training

The training setup is as follows:

We trained the models one at a time with the unordered dataset and then with each of the ordering metrics previously established. All orderings were performed in a reversed and unreversed fashion.

For every order, a new model is initialized. This model is trained using the encoded ordered subset, and using the validation set from SNLI for training evaluation.

These are the notable training arguments used:

- *learning rate*: 0.000002
- *batch size*: 500
- *number of training epochs*: 1
- *weight decay* : 0.01

5.3 Evaluation

After training each model, the overall accuracy is calculated. This is done on the SNLI test set. The results can be found in the next section.

6 Results

6.1 Accuracy scores

The accuracies per subset ordering are as shown in Table 1.

Metric	Normal	Reversed
Original Order	0.8325	0.8320
Jaccard Similarity	0.8313	0.8327
Jaccard Similarity POS	0.8339	0.8293
Height Difference	0.8295	0.8296
Relative Height Difference	0.8325	0.8329
Relative Height Sum	0.8345	0.8348

Table 1: Accuracies of the DistilBERT model fine-tuned with different training orderings

6.2 Discussion

As shown in Table 1 the models trained failed to show a significant difference in accuracy in all separate metrics with all metrics having an accuracy of around 83% including the original ordering. This means that our attempts at creating a CL syllabus failed since training with a specific ordering does not significantly improve accuracy compared to the original ordering. We also attempted to train using 250,000 examples and while seeing an accuracy during training of circa 86%, we found it too computationally intensive for the time we had and also failed to notice any significant difference in accuracy between metrics in the initial runs just like the smaller dataset. Therefore we focused on the smaller training dataset.

However, our accuracy results are still comparable to those in Schluter and Varab (2018) even with the smaller training dataset. This is likely due to the model being used rather than the learning syllabus. For future work, a more sophisticated

model, like BERT or RoBERTa could be used, due to their larger model capacity.

Another option is using the whole dataset instead of only a subset, as this might yield some differences in results.

Furthermore, the metrics that were used focused mostly on the parsed trees of the sentences. Perhaps there are other suitable metrics that could create a better learning curriculum. One example can be using WordNet to see the meanings of words and determine complexity from there. It is also possible to combine all of these metrics into one more complex metric. However, this fell outside the scope of our project and can be done as future work.

7 Conclusion

In conclusion, our research explored whether there are ways to order the training samples in a subset of the SNLI dataset in a way that they reflect increasing levels of difficulty, such that they boost the performance of the model for NLI tasks in comparison to the performance using unorganized training samples. With several experiments including different metrics to compare orderings in the training samples, there has not been found a significant boost in the performance of the model, when comparing it to our original baseline model. A possible reason for this insignificant difference in accuracy between the different metrics, as shown in Table 1, could be the size of our used subsets which were chosen due to computational scarcity. Further research could extend this research by using the whole dataset, implementing different metrics, or using a more sophisticated language model than DistilBERT.

8 Code submission

The code that is used for the experiments, including documentation, can be found on GitHub. This repository also contains the generated subset and code samples in order to easily reproduce our experiments and results.

9 Contributions

This paper and the code implementation used for the experiments were a collaborative effort, with each author contributing an equal amount of time and effort. An overview of the specific contributions can be found in the subsections below.

9.1 Paper Contributions

Yfke Smit:

- Abstract
- Introduction
- Methodology: Subset creation and keeping triplets together
- Experimental setup

Sarah Abdalla:

- Introduction
- Preliminaries: Dataset Description, POS tags + Jaccard Similarity
- Related Work
- Methodology: sections 4.2.1 and 4.2.2
- Conclusion

Tomás Pires Iken:

- Preliminaries: Parsed trees
- Methodology: section 4.2.3
- Results

9.2 Code Contributions

Yfke Smit:

- Subset creation
- Keeping triplets together
- Code cleanup & organization

Sarah Abdalla:

- Tree height metric
- POS tags + Jaccard Similarity metric

Tomás Pires Iken:

- Relative height metric
- Formatting SNLI dataset

10 Acknowledgments

We would like to thank our teacher for the Logic and Language course at Utrecht University this assignment was part of, Lasha Abzianidze, for his help during this project. In addition, we have made use of the provided Logic and Language tools by Lasha Abzianidze for the experiments in this research.

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 41–48, New York, NY, USA. Association for Computing Machinery.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. *CoRR*, abs/1805.02266.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Yucheng Lu, Si Yi Meng, and Christopher De Sa. 2022. A general analysis of example-selection for stochastic gradient descent. In *International Conference on Learning Representations*.

Leonardo Ranaldi, Giulia Pucci, and Fabio Massimo Zanzotto. 2023. Modeling easiness for training transformers with curriculum learning. In *Proceedings of the 14th International Conference on Recent Advances in Natural Language Processing*, pages 937–948, Varna, Bulgaria. INCOMA Ltd., Shoumen, Bulgaria.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Natalie Schluter and Daniel Varab. 2018. When data permutations are pathological: the case of neural natural language inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4935–4939, Brussels, Belgium. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Benfeng Xu, Licheng Zhang, Zhendong Mao, Quan Wang, Hongtao Xie, and Yongdong Zhang. 2020. Curriculum learning for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6095–6104.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78.