Dynamic Sampling Strategies for Multi-Task Reading Comprehension

Ananth Gottumukkala[†] **Dheeru Dua**[†] **Sameer Singh**[†] **Matt Gardner**[‡]
[†]University of California, Irvine, USA
[‡]Allen Institute for Artificial Intelligence, Irvine, California, USA
{agottumu, ddua, sameer}@uci.edu mattg@allenai.edu

Abstract

Building general reading comprehension systems, capable of solving multiple datasets at the same time, is a recent aspirational goal in the research community. Prior work has focused on model architectures or generalization to held out datasets, and largely passed over the particulars of the multi-task learning set up. We show that a simple dynamic sampling strategy, selecting instances for training proportional to the multi-task model's current performance on a dataset relative to its single-task performance, gives substantive gains over prior multi-task sampling strategies, mitigating the catastrophic forgetting that is common in multi-task learning. We also demonstrate that allowing instances of different tasks to be interleaved as much as possible between each epoch and batch has a clear benefit in multi-task performance over forcing task homogeneity at the epoch or batch level. Our final model shows greatly increased performance over the best model on ORB, a recently-released multi-task reading comprehension benchmark.

1 Introduction

Building multi-task reading comprehension systems has received significant attention and been a focus of active research (Talmor and Berant, 2019; Xu et al., 2019). These approaches mostly focus on model architecture improvements or generalizability to new tasks or domains. While these contributions are important, it is also important to explore the optimal way to structure training; as we will show, training on instances from diverse datasets (tasks) means that unlike in a single-task setting, ample instances from each task distribution must be represented during training to properly capture that diversity. We explore 2 fundamental aspects of structuring multi-task training: how many instances are sampled from each task per epoch and how those instances are organized within the epoch. We investigate the importance of this structuring by training a multi-task model on the 8 datasets from ORB (Dua et al., 2019b), a recent multi-task reading comprehension benchmark.

We first explore the sampling distribution over datasets at each epoch: how many instances from each dataset should be used to train. Prior work has typically either made this a uniform distribution over datasets (implicitly favoring smaller datasets), a distribution proportional to the sizes of the datasets (implicitly favoring larger datasets), or some combination of the two. Because these sampling strategies favor some datasets over others, they can lead to catastrophic forgetting in the non-favored datasets. We introduce a *dynamic sampling strategy* that selects instances from a dataset with probability proportional to the gap between its current performance on some metric (like EM or F1 score) and measured single-task performance of the same model on that dataset. By adjusting the sampling distribution over the course of training according to what the model is learning, this method is able to mitigate the catastrophic forgetting that is observed with other sampling strategies.

Next we explore the impact of within-epoch scheduling strategies: once a set of instances has been selected for training, how should they be ordered and batched together? We explore three different strategies: partitioning, homogeneous batches, and heterogeneous batches. We observe a steady increase in performance as instances from different datasets become more and more interleaved within an epoch (less partitioned) and batches are more heterogeneous. This suggests that more variety in batches aids convergence when performing gradient descent steps as opposed to steps using homogeneous batches which only update the model with respect to one task at a time. Partitioning also yields poorer performance since it does not allow the model to see the least recent tasks later in the epoch which leads to catastrophic forgetting on those tasks.

We empirically evaluate these various training strategies on ORB, a recent multi-task reading comprehension benchmark: we take the previous best published model and retrain it using dynamic sampling and heterogeneous batches, leading to a performance increase averaging about 12 points EM and 9 points F1 per task. While we only evaluate on reading comprehension, the methods we present are quite general and can be applied to any multi-task learning setting.

2 Sampling and Scheduling Strategies

We explore two main dimensions along which the instances are ordered in multi-task learning: (1) *instance sampling* from each dataset to get a collection of examples to use for an epoch; and (2) *within-epoch scheduling* of those instances, determining how they should be ordered and batched. The key consideration for these various strategies is avoiding a phenomenon similar to "catastrophic forgetting" (Carpenter and Grossberg, 1988), where performance on a specific dataset in an unbalanced training set can drop dramatically when training moves on from that dataset.

2.1 Instance Sampling

We investigate the following four alternatives for determining how many instances to draw from each dataset for each epoch:

Uniform The simplest way is to uniformly sample instances for each task (Caruana, 1997), which results in an approximately equal number of instances from each dataset per epoch. In practice, this means randomly sampling the same number of training instances from each dataset at each epoch, which will likely be a small subset of all the training instances, as the number of instances in constrained by the smallest dataset. Large datasets will be proportionally underrepresented here.

By Size Alternatively, unbalanced datasets can be dealt with by sampling from each task in proportion to their training set size (e.g. Sanh et al., 2019). However, this approach can result in underfitting small-sized tasks and overfitting large-sized tasks if the ratio between size differences is too extreme.

Uniform→Size [1] This sampling scheme simply has instances sampled uniformly for the first half of training epochs and has instances sampled by training set size for the second half.

Dynamic The prior two methods use a fixed sampling distribution for every epoch of training. We introduce a new, dynamic sampling strategy that aims to focus training on places where it is most needed. For this sampling strategy, we first compute single-task validation metrics for the model that we are training. For each task, we calculate the gap between current multi-task performance and the respective single-task performance and normalize these metric differentials to create a probability distribution. Then, for every epoch after the first (where we use sampling by size), we sample instances by task from this distribution. If performance on a dataset is far from single-task performance, it will get sampled heavily, while datasets that have reached or exceeded single-task performance will get sampled little if at all.[2]

We also experimented with modifying the metric used to calculate the differential. We tested using the 1) validation loss differential, 2) validation EM differential, 3) validation F1 differential, and 4) the sum of the validation EM and F1 differentials (EM+F1 differential). Amongst these, the validation loss for each dataset reaches the single-task loss far quicker than others. This is likely due to the phenomenon that neural networks can overfit to specific loss functions while still benefitting in terms of accuracy (Guo et al., 2017).This explains why the gap in accuracy metrics can be so wide while the loss gap closed within 1 or 2 epochs. Because of this behavior, the loss differentials were all nearly identical in the first few epochs and behavior became very similar to uniform sampling. We finally decided to use EM+F1 differential as this yielded nominally better performance than EM or F1 differential and significantly better performance than loss differential.

2.2 Epoch Scheduling

We explore several different methods for scheduling and batching the instances within an epoch after the set of instances has been sampled:

Partitioned This scheduling strategy partitions the instances in the epoch by task. In other words, the model will never see an instance from a new dataset until all the instances from the current

[1] github.com/mrqa/MRQA-Shared-Task-2019

[2] Sharma and Ravindran (2017) use a related technique in reinforcement learning, though the setup is different.

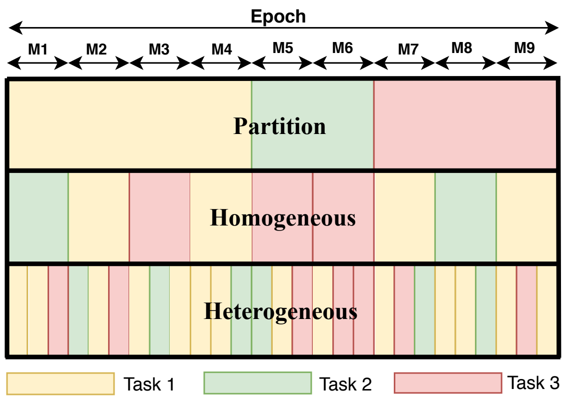

Figure 1: **Illustration of Epoch Scheduling Strategies with Dynamic Sampling.** Instances are sampled dynamically in proportion to exact match accuracy difference of 25%, 10% and 15% for task 1, 2 and 3 respectively. M1, M2, ... M9 depict nine mini-batches in an epoch.

dataset are exhausted. It seems intuitive that this strategy would exacerbate catastrophic forgetting on the tasks it saw least recently, especially when there are a large number of tasks. We include this method simply for completeness.

Homogeneous Batches This scheduling strategy does not force instances into partitions based on the dataset. Instead, instances from each dataset are batched together, then the batches are shuffled.

Heterogeneous Batches This scheduling strategy shuffles all selected instances for the epoch, then batches them together. Each batch could have instances from many different datasets.

Uniform Batches This scheduling strategy is used by the baseline model for the MRQA shared task (Fisch et al., 2019) as well as for the best prior result on ORB. This method places one instance per dataset in each batch (forced heterogeneity) until the smallest dataset runs out of instances. This strategy continues with the remaining datasets, until all datasets are exhausted.

3 Experiments

Setup The eight reading comprehension tasks are from the ORB benchmark (Dua et al., 2019b): DROP (Dua et al., 2019a), DuoRC (Saha et al., 2018), NarrativeQA (Kočisky et al., 2017), NewsQA (Trischler et al., 2017), Quoref (Dasigi et al., 2019), ROPES (Lin et al., 2019), SQuAD (Rajpurkar et al., 2016), and SQuAD 2.0 (Rajpurkar et al., 2018). We use the NABERT[3] (Numerically Augmented BERT) model with an additional reasoning type to allow "No Answer" as an answer to accommodate the SQuAD 2.0 dataset which has about 40,000 "No Answer" questions. Each training session lasted 30 epochs with 50,000 instances sampled per epoch. Three training sessions were conducted per sampling method and the EM and F1 scores shown are averaged over those three sessions. Note that NarrativeQA is evaluated using only ROUGE F1 score. Due to GPU memory constraints, we are limited to a batch size of 4, so we are unable replicate the *Uniform Batches* configuration of MRQA (requires a batch size of 8 to fit 1 instance from each of the 8 datasets).

Dataset	**Train Size**	**Dev Size**
Small		
Quoref	19,392	2,407
ROPES	10,924	1,688
Medium		
DuoRC	54,746	12,224
NarrativeQA	32,717	3,393
Large		
DROP	77,394	9,530
NewsQA	92,543	5,154
SQuAD1.1	87,596	10,570
SQuAD2.0	130,310	11,864

Table 1: Open Reading Benchmark (ORB) Datasets

Sampling Strategies Table 2 shows the effectiveness of the sampling techniques discussed above. Uniform sampling yields a very mediocre performance for 7 datasets but significantly underperforms on SQuAD 2.0, which is likely not getting enough representation each epoch for its unique no-answer questions. Sampling by size yields mediocre performances for 7 datasets but underperforms on ROPES, which is easily the smallest dataset and therefore gets undersampled. However, performance on Quoref, the second smallest dataset, is still relatively high, which might be explained by its SQuAD-style questions. Exposure to SQuAD, one of the largest datasets, likely benefits performance on Quoref as well. Interestingly, uniform sampling followed by size sampling slightly alleviates the problems from the individual sampling methods but also slightly underforms

[3]https://github.com/raylin1000/drop_bert

Method	Average		Quoref		ROPES		DuoRC		NarrQA		SQuAD		SQuAD2		DROP		NewsQA	
	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1
Single Task	-	-	53.0	58.6	67.5	72.1	23.3	30.8	-	50.3	57.5	73.5	66.0	69.6	57.1	54.4	35.3	49.8
Uniform	49.2	55.8	56.9	61.5	69.7	74.3	23.4	32.1	-	53.1	69.3	78.0	38.1	42.9	51.8	54.4	35.0	49.9
By Size	50.0	56.3	53.7	57.7	62.7	68.1	23.3	31.6	-	52.4	65.8	74.1	58.1	63.0	52.0	54.5	34.6	49.1
Uni$\rightarrow$Size	49.7	56.5	55.8	60.0	68.8	73.8	23.2	32.0	-	53.0	52.0	63.7	63.4	67.4	49.7	52.2	35.0	49.8
Dynamic	**51.7**	**58.1**	56.3	60.4	65.1	71.9	23.1	31.5	-	52.9	66.3	74.7	63.2	67.7	53.8	56.3	34.5	49.2

Table 2: Effect of using different instance sampling strategies with heterogeneous batch scheduling

Method	Average		Quoref		ROPES		DuoRC		NarrQA		SQuAD		SQuAD2		DROP		NewsQA	
	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1
Partition	46.1	53.2	50.7	55.3	58.1	65.4	22.1	30.7	-	50.9	67.0	76.6	36.5	41.6	55.3	58.2	32.0	47.4
Homo	48.8	54.7	53.3	56.8	61.5	66.6	21.6	29.6	-	49.9	63.7	71.7	56.0	60.6	51.8	54.1	33.5	48.2
Hetero	**51.7**	**58.1**	56.3	60.4	65.1	71.9	23.1	31.5	-	52.9	66.3	74.7	63.2	67.7	53.8	56.3	34.5	49.2

Table 3: Effect of using different epoch scheduling strategies with dynamic sampling

Method	Average		Quoref		ROPES		DuoRC		NarrQA		SQuAD		SQuAD2		DROP		NewsQA	
	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1	EM	F_1
ORB	34.4	42.1	35.0	44.7	31.1	37.3	25.4	34.1	-	36.6	67.3	77.7	32.8	38.0	20.2	23.6	29.2	44.6
Dynamic	**47.6**	**54.5**	59.4	63.9	36.5	44.8	23.0	31.5	-	52.0	66.3	74.7	61.2	65.7	51.9	54.2	34.7	49.1

Table 4: Results on ORB test sets.

on DROP. Finally, dynamic sampling achieves the highest average performance and fully cures both problems mentioned above since each epoch, the sampling distribution can be adjusted based on which datasets perform poorly. The previous sampling methods have static sampling distributions, so these adjustments are impossible.

Scheduling Strategies Table 3 show that heterogeneous batches during sampling leads to the best multi-task performance, and performance steadily decreases as instance grouping becomes more and more homogenized with respect to the dataset.

ORB Evaluation Finally, Table 4 shows that our model trained with dynamic sampling and heterogeneous batches significantly outperforms the previous ORB state-of-the-art NABERT baseline model (submitted on 11/12/2019 on the leaderboard site[4]).

[4]https://leaderboard.allenai.org/orb/submissions/public

4 Conclusions

Our goal was to investigate which instance sampling method and epoch scheduling strategy gives optimal performance in a multi-task reading comprehension setting. The results suggest that dynamic sampling—sampling instances from each task based on their respective metric differentials—is a fruitful direction to explore for improving performance. We also show that interleaving instances from different tasks within each epoch and forming heterogeneous batches is crucial for optimizing multi-task performance. It is also worth noting that for the DuoRC, NarrativeQA, SQuAD, and Quoref datasets there are cases where the multi-task model outperforms the single-task model. This suggests that for specific cases, we observe an effect similar to data augmentation (like exposure to SQuAD benefitting QuoREF performance as mentioned above) but this needs to be explored further. We hope that future work experiments further with dynamic sampling such as by modifying the metric (e.g., using BLEU or ROUGE score if applicable) and/or modifying other values like number of instances per epoch based on performance metrics (not only does this effectively change learning rate, but it would also allow the model to update the sampling distribution more or less frequently).

Acknowledgements

This work was supported in part by funding from Allen Institute of Artificial Intelligence, in part by Amazon, and in part by the National Science Foundation (NSF) grant #CNS-1730158.

References

Gail A Carpenter and Stephen Grossberg. 1988. The art of adaptive pattern recognition by a self-organizing neural network. *Computer*, 21(3):77–88.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Pradeep Dasigi, Nelson Liu, Ana Marasovic, Noah Smith, and Matt Gardner. 2019. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. In *EMNLP*.

D. Dua, Y. Wang, P. Dasigi, G. Stanovsky, S. Singh, and M. Gardner. 2019a. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *North American Association for Computational Linguistics (NAACL)*.

Dheeru Dua, Ananth Gottumukkala, Alon Talmor, Sameer Singh, and Matt Gardner. 2019b. Orb: An open reading benchmark for comprehensive evaluation of machine reading comprehension. In *Proceedings of the Second Workshop on Machine Reading for Question Answering*, pages 147–153.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. Mrqa 2019 shared task: Evaluating generalization in reading comprehension. *arXiv preprint arXiv:1910.09753*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks.

T. Kočisky, J. Schwarz, P. Blunsom, C. Dyer, K. M. Hermann, G. Melis, and E. Grefenstette. 2017. The NarrativeQA reading comprehension challenge. *arXiv preprint arXiv:1712.07040*.

Kevin Lin, Oyvind Tafjord, Peter Clark, and Matt Gardner. 2019. Reasoning over paragraph effects in situations. *arXiv preprint arXiv:1908.05852*.

P. Rajpurkar, R. Jia, and P. Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Association for Computational Linguistics (ACL)*.

P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*.

A. Saha, R. Aralikatte, M. Khapra, and K. Sankaranarayanan. 2018. Duorc: Towards complex language understanding with paraphrased reading comprehension. In *Association for Computational Linguistics (ACL)*.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6949–6956.

Sahil Sharma and Balaraman Ravindran. 2017. Online multi-task learning using active sampling.

Alon Talmor and Jonathan Berant. 2019. Multiqa: An empirical investigation of generalization and transfer in reading comprehension. *Association for Computational Linguistics*.

A. Trischler, T. Wang, X. Yuan, J. Harris, A. Sordoni, P. Bachman, and K. Suleman. 2017. NewsQA: A machine comprehension dataset. In *Workshop on Representation Learning for NLP*.

Yichong Xu, Xiaodong Liu, Yelong Shen, Jingjing Liu, and Jianfeng Gao. 2019. Multi-task learning with sample re-weighting for machine reading comprehension. *North American Chapter of the Association for Computational Linguistics*.