ADAPTIVE MULTI-CORPORA LANGUAGE MODEL TRAINING FOR SPEECH RECOGNITION

Yingyi Ma, Zhe Liu, Xuedong Zhang

Meta AI, Menlo Park, CA, USA

ABSTRACT

Neural network language model (NNLM) plays an essential role in automatic speech recognition (ASR) systems, especially in adaptation tasks when text-only data is available. In practice, an NNLM is typically trained on a combination of data sampled from multiple corpora. Thus, the data sampling strategy is important to the adaptation performance. Most existing works focus on designing static sampling strategies. However, each corpus may show varying impacts at different NNLM training stages. In this paper, we introduce a novel adaptive multi-corpora training algorithm that dynamically learns and adjusts the sampling probability of each corpus along the training process. The algorithm is robust to corpora sizes and domain relevance. Compared with static sampling strategy baselines, the proposed approach yields remarkable improvement by achieving up to relative 7% and 9% word error rate (WER) reductions on in-domain and out-of-domain adaptation tasks, respectively.

Index Terms— automatic speech recognition, language model, adaptation, multi-corpora, adaptive training

1. INTRODUCTION

Language models have been commonly used to promote automatic speech recognition (ASR) performance through first-pass fusion [1, 2, 3] or second-pass rescoring [4, 5, 6, 7]. Recent studies show that neural network based language model (NNLM) obtains significantly stronger performance than traditional n-gram models given its better capability of modeling long-range dependency [8, 9, 10, 11, 12].

Typically, an external language model can be trained separately from ASR models, thus bringing further flexibility to its choice of training data and training strategy. Since only text data is required for training any language model, it provides enormous advantages on leveraging external language models on adaptation tasks for ASR applications where there is a mismatch between source and target domains.

The proper performance of a language model highly depends on the quality and quantity of training data in any adaptation task. However, it is hard to guarantee this especially when the target domain is lack of adaptation data [13, 14], or such data is not fully accessible due to privacy pursuit [15, 16, 17, 18]. Alternatively, we can resort to text data from multiple corpora. Some corpora may be in a similar domain with the target while some may not. Hence, how to improve the performance of a language model by effectively leveraging large-scale multiple corpora becomes a critical research topic.

Training a single n-gram language model on multiple corpora is relatively straightforward. This is usually done by training an n-gram model for each corpus and then interpolating them to form a final model. The interpolation weights are usually optimized over a validation set from the target domain. The linearity of the parameters in n-gram language models grants direct interpolation, while it is hard to see an easy extension to non-linear NNLMs. Instead of model interpolation, mixing data from diverse corpora fits better for mini-batch fashion training. However, we found that the strategies for sampling data from multiple corpora lack study. A few works explore data mixing methods based on the relevance between each corpus and the target domain [19, 20], nonetheless, all these methods employ static sampling strategies, where each corpus's sampling probability is precomputed and fixed along the training process. Recent studies show that the order of feeding corpus into training is important for boosting the model performance [21, 22, 23], indicating that the sampling probability of each corpus may need to be adjusted dynamically.

Motivated by the challenges and insights above, we propose a novel adaptive multi-corpora NNLM training algorithm. We adjust the sampling probabilities based on the estimated contribution that each corpus can devote in the following training stage. Specifically, at the beginning of each epoch, we fine-tune the current model on each corpus separately, then interpolate the fine-tuned models and optimize the interpolation weights over the validation dataset of the target domain. The optimized weights will then be leveraged as the sampling probabilities over multiple corpora in the current training epoch. Our method can work with any arbitrary sized corpora and adaptively take into account the importance of each corpus along the entire training process. We conduct experiments on in-domain and out-of-domain adaptation scenarios, with results indicating that the proposed adaptive sampling probability adjustment is effective in improving the performance of NNLM for ASR applications.

2. RELATED WORKS

Although various works have studied data sampling and weighting strategies across language model and deep learning training tasks [24, 25, 26], most of them are applied to the sample level other than the corpus or dataset level. Learning to reweight samples usually requires the computation of second-order derivatives in every backward pass, which leads to high complexity and thus could be heavy-loaded for multi-corpora training.

As another line of research, recent works have shown that the order of corpora plays a critical role in training language models [21, 22, 23]. Hence, abundant metrics for measuring the relevance over corpora have been designed, and curriculum learning approaches in lieu of data mixing strategies have also been proposed. However, a well-designed adaptive data mixing strategy naturally incorporates the importance of corpora order during the training process.

Towards joint training with multiple corpora, multi-task based training strategies are introduced [27, 28, 29]. These works train a multi-head language model with several shared hidden layers and corpora-specific output layers. In adaptation tasks, they learn the interpolation weights for combining corpora-specific layers and result in a single NNLM. However, the parameter size and inference

latency of such model would grow with the number of corpora, making this type of approaches less attractive in practical applications.

Inspired by the interpolation of n-gram models, authors in [19] propose to train an n-gram model on each corpus and optimize their interpolation weights over a validation set. Then during the NNLM training, minibatches are stochastically constructed by drawing samples from each corpus with probability according to the interpolation weights. These weights are only learned once from n-gram models and fixed over the entire NNLM training process. Unlike this approach, our proposed method adaptively optimizes the sampling probability per corpus during the training process.

3. METHODS

Consider an NNLM adaptation task where we are given K different training corpora $D_1, \ldots, D_K$ with each $D_k = \{x_i^{(k)}\}_{i=1}^{N_k}$ consisting of N_k records, as well as a validation corpus $D_\tau = \{y_i\}_{i=1}^{N_\tau}$ from the target domain. The goal is to train a model with records sampled from the K corpora that can achieve optimized performance on the target domain. Note that N_τ is usually a much smaller number compared with these N_k's, and the target domain could be close-in-domain with a few ones of these K corpora or a completely different domain.

Towards this goal, it is crucial to develop a sampling strategy for mixing multiple corpora during model training. We propose to adjust the sampling probability per corpus to fit best for each training stage and optimize it over the validation set dynamically. Specifically, the NNLM training is divided into an upper-level task and a lower-level task. Optimizing the data sampling probability across multiple corpora is regarded as the upper-level task, while the lower-level task is in charge of learning model parameters of NNLM given current sampling probabilities. The upper-level task and lower-level task are conducted in an alternating manner until model convergence.

We first introduce a generic training framework using mixed data from multiple corpora in the following subsection.

3.1. Mixed Data Training

Define $\mathcal{D} = \mathcal{M}(D_1, \ldots, D_K|W)$ as a mixed data distribution over corpora $D_1, \ldots, D_K$ such that

$$\forall x \sim \mathcal{D} : P(x \in D_k) = w_k, \tag{1}$$

$$P(x = x_i^{(k)} | x \in D_k) = \frac{1}{N_k},\ \forall k \in [1..K] \tag{2}$$

where $W = (w_1, \ldots, w_K)$ is the sampling probability over corpora and $\Sigma_{k=1}^K w_k = 1$. This mixed data distribution will sample data from corpus D_k with probability w_k. That is, at each training iteration (i.e. model update), we sample a minibatch of training records from $\mathcal{D}$ with roughly w_k portion of data coming from corpus D_k.

Given the mixed data distribution $\mathcal{D}$, an NNLM θ can be trained by minimizing the negative log-likelihood of samples:

$$\mathcal{L}_{train}(\theta|\mathcal{D}) = -\Sigma_{x\sim\mathcal{D}} \log P_\theta(x) \tag{3}$$

where $P_\theta(x)$ is the model estimated probability of observing x as a product of each token's probability given its preceding tokens.

As each corpus may show varying impacts on the target domain at different training stages, we will adjust the sampling probability W over multiple corpora before each training epoch of NNLM, with details described in the next subsection.

3.2. Sampling Probability Optimization

Adjusting the sampling probability W aims to optimize the contribution of each corpus in the following training. Thus, we first need to measure the effects of continuing training with each corpus, and then adapt the sampling probability accordingly. To this end, we propose to fine-tune the current NNLM using each corpus solely, after which interpolate K fine-tuned models and optimize the interpolation weights over the validation set of the target domain. The learned weights will serve as the sampling probability in the next. We break down this process into the two steps below.

3.2.1. The fine-tuning step

Define a fine-tuning operator as

$$FT(\theta, D, S) \mapsto \theta_{FT} \tag{4}$$

which fine-tunes input model θ on corpus D for S iterations. To fairly measure the contribution each corpus could devote in the following training process, we fine-tune the current model θ on each corpus for the same number of iterations S. Then, we will obtain K different models $\theta_1, \ldots, \theta_K$:

$$\theta_k = FT(\theta, D_k, S),\ \forall k \in [1..K] \tag{5}$$

The fine-tuning step is a continued training from the current model θ per corpus and conducted at the beginning of each epoch.

3.2.2. The interpolation weight optimization step

Consider an interpolation weight optimization operator:

$$WO(\Theta, D_\tau) \mapsto W \tag{6}$$

where $\Theta = (\theta_1, \ldots, \theta_K)$ is a collection of K fine-tuned models. The WO operator will optimize the model interpolation weights $W = (w_1, \ldots, w_K)$ over the performance on D_τ and then output the optimized weights. Specifically, the WO operator can be defined as follows:

$$W = \underset{w_1,\ldots,w_K}{\operatorname{argmin}} -\Sigma_{y_i \in D_\tau} \log(\Sigma_{k=1}^K w_k \cdot P_{\theta_k}(y_i)) \tag{7}$$

$$s.t. \quad w_k \in [0, 1],\ \forall k \in [1..K] \tag{8}$$

$$\Sigma_{k=1}^K w_k = 1 \tag{9}$$

By substituting w_K with $1 - w_1 - \ldots - w_{K-1}$, the optimization problem above can be solved through either gradient descent (adopted in this work) or an expectation-maximization (EM) algorithm.

As the interpolation weights well reflect the importance of each corpus in the following training process, we can make use of them as the sampling probabilities over corpora for the current NNLM training epoch (1)-(3).

3.3. The Adaptive Multi-Corpora Training Framework

To chain all the steps we have introduced, in Algorithm 1, we present the adaptive sampling framework for multi-corpora NNLM training. The proposed method adaptively determines the sampling probability per corpus during the training process. Higher probabilities are assigned to the ones who will likely play more important roles towards improving the performance on target domain in certain training epochs, and vice versa. In the scenarios where the order of feeding corpora into training is vital, the proposed algorithm can automatically choose their relative orders by assigning higher sampling probability to a corpus when its turn is coming.

Algorithm 1 Adaptive multi-corpora training

Input: K training corpora $D_1, \ldots, D_K$, validation dataset of target domain D_τ, number of fine-tuning iterations S
Initialize NNLM θ^0
for epoch $t = 1, 2, \ldots, T$ **do**
  for each corpus $k = 1, 2, \ldots, K$ **do**
    $\theta_k^t = FT(\theta^{t-1}, D_k, S)$
  end for
  $W^t = WO((\theta_1^t, \ldots, \theta_K^t), D_\tau)$
  Construct $\mathcal{D}^t = \mathcal{M}(D_1, \ldots, D_K | W^t)$
  Learn θ^t through training on $\mathcal{D}^t$ for one epoch: $\text{argmin}_{\theta^t} \left(-\Sigma_{x \sim \mathcal{D}^t} \log P_{\theta^t}(x)\right)$
end for
Output: θ^T

Oftentimes, corpora available for training could be of diverse sizes due to data scarcity. The proposed algorithm is unfastidious in the various sizes of corpora. It performs the same number of training iterations when fine-tuning on each corpus, and if some corpora are of petite size but turn out to be frequently sampled, it simply reflects the importance of these corpora in certain training stages.

In Algorithm 1, the frequency of optimizing sampling probabilities is once per training epoch. It can be flexibly adjusted in different applications in practice. For example, we can adapt the sampling probabilities every Q training iterations instead and the rest components in the algorithm should naturally follow.

It is noteworthy that the proposed method works differently than the bi-level optimization framework [30, 31] in the sense that the sampling probabilities learned at the end of training process only represents its optimality for the final training stage and can not be regarded as optimal over the entire training process. The proposed algorithm dynamically picks the optimal sampling probabilities at each training stage.

Since the proposed method requires fine-tuning on each corpus before each training epoch to determine the sampling probabilities over corpora, it needs additional $K \cdot S \cdot T$ training iterations than the conventional NNLM training with static sampling probabilities. Notice that the fine-tuning process on each corpus are independent with each other and thus can be conducted in parallel for better efficiency.

4. EXPERIMENTS

4.1. Datasets

Table 1 summarizes the corpora used in our experiments as well as their train, validation, and test splits. Among them, we have

- Publicly available datasets, including Fisher speech corpus (*fisher*), Wall Street Journal corpus (*wsj*), and Wikitext-103 (*wiki*). For *fisher* corpus, we only utilize text training data in this study and each record represents a conversation. For *wsj* corpus, we use *nov93dev* as the validation set and *nov92* as the test set;
- In-house datasets, which contain video corpus sampled from public social media videos (*video*), and three conversational speech corpora with different topics (*conv1*, *conv2*, *conv3*) collected using mobile devices through crowd-sourcing from a data supplier for ASR. All these datasets are de-identified before transcription; all transcribers and researchers do not have access to any user-identifiable information.

Table 1. Summary of data splits from multiple corpora.

Corpora	Splits		
	Train (#text records)	Validation (#utts)	Test (#utts)
conv1	138K	1K	4K
conv2	2000K	-	-
video	1100K	-	-
wiki	840K	-	-
fisher	12K	-	-
conv3	-	1K	12K
wsj	-	0.5K	0.3K

4.2. Setups

We consider two adaptation scenarios in this study

- *In-domain adaptation*, where one of the training corpora is in the same domain with the target;
- *Out-of-domain adaptation*, none of the training corpora being in the same domain with the target.

For each setting, NNLMs are trained on multiple corpora and integrated with an ASR model via first-pass shallow fusion. We then evaluate the performance of trained NNLMs on the test sets of target domain in terms of perplexity (PPL) and word error rate (WER).

Each NNLM contains an embedding layer of dimension 300, and 2 LSTM layers with 1500 hidden nodes, which has around 15 million parameters in total. The ASR is a RNN-T model with the Emformer encoder [32], LSTM predictor, and a joiner. It has around 80 million parameters and is trained from scratch using the train split of LibriSpeech ASR corpus [33].

The following multi-corpora NNLM training methods are compared in our experiments:

- `uniform-weight`, which assigns the same sampling probability to each training corpus. Note that this method is close to the "data merging" approach where the models are trained on merged corpora, but simply merging data altogether fails to balance the size of each training corpus;
- `ngram-opt-weight`, the method presented in [19], where an n-gram model is trained on each corpus, and the optimized interpolation weights (with respect to the validation set) from these n-gram models are used as the fixed sampling probabilities over multiple corpora during NNLM training;
- `adaptive-weight`, our proposed method in Algorithm 1.

4.3. Results

4.3.1. In-domain adaptation

We first conduct a set of experiments on learning NNLMs with the train split of one corpus (*conv1*), two corpora (*conv1+conv2*), or three corpora (*conv1+conv2+video*). The validation and test sets of *conv1* are considered as the ones from the target domain. Hence, these experiments are regarded as in-domain adaptation tasks since the train split of *conv1* also appears in the training corpora.

Table 2 demonstrates the PPL and WER evaluation results on the test set of *conv1*. We can observe that the proposed adaptive training algorithm achieves the best performance in both scenarios of two corpora training and three corpora training. Compared with

`uniform-weight` and `ngram-opt-weight` methods, our approach results in relative 3%-5% and 5%-7% WER reductions, respectively. It is also expected that leveraging more corpora in the training set generally improves the NNLM quality.

Table 2. PPL and WER results on *conv1* test set.

Train Corpora	NNLM Training Method	PPL	WER
n/a	`without-NNLM`	-	24.60
conv1	`uniform-weight`	54.2	20.87
conv1+conv2	`uniform-weight`	43.5	19.75
	`ngram-opt-weight`	48.5	20.15
	`adaptive-weight`	**38.8**	**18.82**
conv1+conv2 +video	`uniform-weight`	36.8	19.20
	`ngram-opt-weight`	43.8	19.63
	`adaptive-weight`	**32.9**	**18.65**

4.3.2. *Out-of-domain adaptation*

Here, NNLMs are trained on three corpora (*conv1+conv2+video*) or five corpora (*conv1+conv2+video+wiki+fisher*), and evaluated on two target domains, *conv3* and *wsj*. Note that each target domain is different from any of the domains in the training corpora. Thus, this study is considered as out-of-domain adaptation.

The PPL and WER evaluation results on the test sets of *conv3* and *wsj* are presented in Table 3 and Table 4, respectively. Similar to our previous findings, the introduced `adaptive-weight` method outperforms all other methods consistently. Compared with `uniform-weight` approach, our method obtains relative 5%-9% WER reductions on *conv3* test set and 6%-9% WER reductions on *wsj* test set. Compared with `ngram-opt-weight` approach, our method achieves relative 3% WER reductions on *conv3* test set and 2%-5% WER reductions on *wsj* test set.

Table 3. PPL and WER results on *conv3* test set.

Train Corpora	NNLM Training Method	PPL	WER
n/a	`without-NNLM`	-	24.79
conv1+conv2 +video	`uniform-weight`	65.6	18.95
	`ngram-opt-weight`	60.2	18.55
	`adaptive-weight`	**49.7**	**17.98**
conv1+conv2+ video+wiki +fisher	`uniform-weight`	63.2	18.92
	`ngram-opt-weight`	48.2	17.91
	`adaptive-weight`	**41.5**	**17.34**

Table 4. PPL and WER results on *wsj* test set.

Train Corpora	NNLM Training Method	PPL	WER
n/a	`without-NNLM`	-	10.96
conv1+conv2 +video	`uniform-weight`	91.2	9.82
	`ngram-opt-weight`	80.8	9.39
	`adaptive-weight`	**65.2**	**8.95**
conv1+conv2+ video+wiki +fisher	`uniform-weight`	78.2	9.44
	`ngram-opt-weight`	66.9	9.10
	`adaptive-weight`	**62.4**	**8.86**

4.4. Analysis

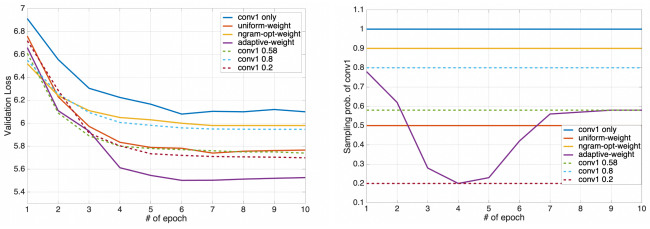
To provide more insights on how the proposed adaptive training algorithm works, we conduct additional analysis on *conv1* in-domain adaptation task with two training corpora (*conv1+conv2*), corresponding to row 4 to 6 in Table 2.

We present the training progress as follows: in Figure 1(a), we compare the validation loss along the training progress of multiple methods with different sampling strategies. Their corresponding sampling probability for the in-domain training corpus *conv1* is shown in Figure 1(b).

Seen from Figure 1(a), the proposed approach converges to a much lower validation loss. Without the use of out-of-domain data, training with in-domain data of *conv1* only performs the worst since the train split of *conv1* is relatively small. Similarly, leveraging the n-gram interpolation weights as the sampling probabilities makes it hard to perform well, because the n-gram model trained on in-domain adaptation set tends to receive a much higher interpolation weight. Assigning such a high probability to a small adaptation set of *conv1* results in insufficient use of the other corpus.

On the other hand, the proposed `adaptive-weight` method can implicitly take into account the size and importance of each corpus while adjusting the sampling probability during the training process. According to the sampling probability curve presented in Figure 1(b), its training process can be split into three stages. In the first two epochs, the model mainly focuses on learning from the in-domain adaptation set *conv1*. Starting from epoch three, continuing learning heavily from the in-domain corpus may lead to overfitting or early convergence. The model thus puts more efforts on learning from the out-of-domain corpus *conv2* until epoch six. After that, the learning rate becomes relatively smaller. The model then seeks a balanced weight for both corpora.

For more comparison, we train additional models by using the starting sampling probability (*conv1* 0.8), lowest probability (*conv1* 0.2), or last probability (*conv1* 0.58) along the training progress of `adaptive-weight`, and assigning it as the static sampling probability during the training. However, none of these methods performs better or even close to the proposed method, further highlighting the significance of an adaptively adjustable data sampling strategy.



(a) (b)

Fig. 1. Training progress for various sampling strategies on *conv1* adaptation task; (a): validation loss versus training epoch; (b): sampling probability for *conv1* versus training epoch.

5. CONCLUSION

In this work, we introduce a novel adaptive multi-corpora training algorithm to improve the performance of NNLM for ASR applications. The proposed approach provides an adaptive data sampling strategy to effectively leverage each corpus along the training process. Experiment results show prominent WER improvement for both in-domain and out-of-domain adaptation tasks. Future work might include extending the presented multi-corpora algorithm to end-to-end ASR model training.

6. REFERENCES

[1] Anjuli Kannan, Yonghui Wu, Patrick Nguyen, Tara N Sainath, Zhijeng Chen, and Rohit Prabhavalkar, "An analysis of incorporating an external language model into a sequence-to-sequence model," in *ICASSP*, 2018.

[2] Suyoun Kim, Yuan Shangguan, Jay Mahadeokar, Antoine Bruguier, Christian Fuegen, Michael L Seltzer, and Duc Le, "Improved neural language model fusion for streaming recurrent neural network transducer," in *ICASSP*, 2021.

[3] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *ICASSP*, 2016.

[4] Xunying Liu, Yongqiang Wang, Xie Chen, Mark JF Gales, and Philip C Woodland, "Efficient lattice rescoring using recurrent neural network language models," in *ICASSP*, 2014.

[5] Hainan Xu, Tongfei Chen, Dongji Gao, Yiming Wang, Ke Li, Nagendra Goel, Yishay Carmiel, Daniel Povey, and Sanjeev Khudanpur, "A pruned RNNLM lattice-rescoring algorithm for automatic speech recognition," in *ICASSP*, 2018.

[6] Ke Li, Daniel Povey, and Sanjeev Khudanpur, "A parallelizable lattice rescoring strategy with neural language models," in *ICASSP*, 2021.

[7] Kazuki Irie, Albert Zeyer, Ralf Schlüter, and Hermann Ney, "Language modeling with deep transformers," in *ICASSP*, 2019.

[8] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur, "Recurrent neural network based language model.," in *Interspeech*, 2010.

[9] Xie Chen, Xunying Liu, Mark JF Gales, and Philip C Woodland, "Improving the training and evaluation efficiency of recurrent neural network language models," in *ICASSP*, 2015.

[10] Hainan Xu, Ke Li, Yiming Wang, Jian Wang, Shiyin Kang, Xie Chen, Daniel Povey, and Sanjeev Khudanpur, "Neural network language modeling with letter-based features and importance sampling," in *ICASSP*, 2018.

[11] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.

[12] Alex Graves, "Sequence transduction with recurrent neural networks," in *ICML Workshop on Representation Learning*, 2012.

[13] Jerome R Bellegarda, "Statistical language model adaptation: review and perspectives," *Speech communication*, vol. 42, no. 1, 2004.

[14] Ke Li, Zhe Liu, Tianxing He, Hongzhao Huang, Fuchun Peng, Daniel Povey, and Sanjeev Khudanpur, "An empirical study of transformer-based neural language model adaptation," in *ICASSP*, 2020.

[15] Ankur Gandhe, Ariya Rastrow, and Bjorn Hoffmeister, "Scalable language model adaptation for spoken dialogue systems," in *SLT*, 2018.

[16] Zhe Liu, Ke Li, Shreyan Bakshi, and Fuchun Peng, "Private language model adaptation for speech recognition," *arXiv preprint arXiv:2110.10026*, 2021.

[17] Dhruv Guliani, Françoise Beaufays, and Giovanni Motta, "Training speech recognition models with federated learning: A quality/cost framework," in *ICASSP*, 2021.

[18] Xiaodong Cui, Songtao Lu, and Brian Kingsbury, "Federated acoustic modeling for automatic speech recognition," in *ICASSP*, 2021.

[19] Anirudh Raju, Denis Filimonov, Gautam Tiwari, Guitang Lan, and Ariya Rastrow, "Scalable multi corpora neural language models for ASR," in *Interspeech*, 2019.

[20] Holger Schwenk and Jean-Luc Gauvain, "Training neural network language models on very large corpora," in *EMNLP*, 2005.

[21] Ameeta Agrawal, Suresh Singh, Lauren Schneider, and Michael Samuels, "On the role of corpus ordering in language modeling," in *ACL Workshop on Simple and Efficient Natural Language Processing*, 2021.

[22] Ernie Chang, Hui-Syuan Yeh, and Vera Demberg, "Does the order of training samples matter? improving neural data-to-text generation with curriculum learning," in *European Chapter of ACL*, 2021.

[23] Xuebo Liu, Houtim Lai, Derek F Wong, and Lidia S Chao, "Norm-based curriculum learning for neural machine translation," in *ACL*, 2020.

[24] Jared Fernandez and Doug Downey, "Sampling informative training data for RNN language models," in *ACL Student Research Workshop*, 2018.

[25] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun, "Learning to reweight examples for robust deep learning," in *ICML*, 2018.

[26] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng, "Meta-weight-net: Learning an explicit mapping for sample weighting," in *NeurIPS*, 2019.

[27] Zoltán Tüske, Kazuki Irie, Ralf Schlüter, and Hermann Ney, "Investigation on log-linear interpolation of multi-domain neural network language model," in *ICASSP*, 2016.

[28] Ashwin Geet d'Sa, Irina Illina, Dominique Fohr, and Awais Akbar, "Exploration of multi-corpus learning for hate speech classification in low resource scenarios," in *TSD*, 2022.

[29] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao, "Multi-task deep neural networks for natural language understanding," in *ACL*, 2019.

[30] Simon Jenni and Paolo Favaro, "Deep bilevel learning," in *ECCV*, 2018.

[31] Risheng Liu, Jiaxin Gao, Jin Zhang, Deyu Meng, and Zhouchen Lin, "Investigating bi-level optimization for learning and vision from a unified perspective: A survey and beyond," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[32] Yangyang Shi, Yongqiang Wang, Chunyang Wu, Ching-Feng Yeh, Julian Chan, Frank Zhang, Duc Le, and Mike Seltzer, "Emformer: Efficient memory transformer based acoustic model for low latency streaming speech recognition," in *ICASSP*, 2021.

[33] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *ICASSP*, 2015.