BTP2 Presentation — Code-switched NLP + NN Optimization

Richeek Das (190260036), Sahasra Ranjan (190050102)

IIT Bombay

May 2023

MLM Pretraining

- Paper accepted at ACL 2023
- Title: Improving Pretraining Techniques for Code-Switched NLP
- Abstract:
  - Explores different masked language modeling pretraining techniques for code-switched text
  - Exploit switch-point information to restrict maskable tokens to learn better representations for the critical tokens in code-switched text.
  - Introduces MLM variant with residual connection to earlier layer in pretrained model for improved performance on downstream tasks
  - Experiments on QA and SA tasks with Hindi-English, Spanish-English, Tamil-English, and Malayalam-English yield relative improvements up to 5.8 and 2.7 F1 scores, respectively, compared to standard pretraining techniques

Switch-MLM

- Switch-point is informed by the transitions between the languages in the code-switched sentence.
- For the following Hindi-English CS sentence: "Laptop Mere Bag Me Rakha Hai", the language ids are: "EN HI EN HI HI HI". And so the switch point boundary words are "Laptop", "Mere", "Bag", "Me".
- We select only these words to be masked during the MLM.
- The only limitation with this approach is that it requires access to LID tagged dataset or a LID tagger for CS sentences. We introduce Freq-MLM to address this issue.

Freq-MLM

- We assign LID tags to tokens based on the relative frequencies obtained from monolingual corpora of the component languages.
- We use different techniques based on the availability of corpus and vocabulary of the component languages:
  - **NLL**: We define nll_X for a word as the negative log likelihood of the word in the corpora of language X. We then compare nll_{L1} and nll_{L2} for component language L1 and L2 to assign correct LID tag to the word.
  - **X-hit**: We check the availability of the word in the vocabulary on the languages and assign the correct LID accordingly. This technique is very specific to the language pairs.

Architectural Modifications

- We work on the mBERT (or XLM-R) model to begin with. We then add residual connections and an auxiliary loss to exploit the switch point information of the CS text.
- **Residual Connections:** Priod studies suggests that the language information could be stored in lower or middle layers. We introduce a simple residual connection from one of these layers to the last layer and add it with some dropout.
- **Auxiliary Loss:** We further encourage the thesis of language information being stored at the lower or middle layers and use embeddings from one of these layers to have a modified loss equation.

Experimental Setup

- We test our hypothesis on Hindi-English, Spanish-English, Malayalam-English and Tamil-English code switched language pairs.
- We use the GLUECoS benchmark to evaluate our models for Sentiment Analysis (SA) and Question Answering (QA) tasks.
- Finding read code switched dataset was one of the most challenging task. We aggregated real CS texts from multiple sources to create a pretraining corpora for HI-EN, ES-EN, TA-EN, ML-EN CS texts consisting of 185k, 66k, 118k and 34k sentences, respectively.

Results

	Method	QA HI-EN F1 (20 epochs)	QA HI-EN F1 (30 epochs)	QA HI-EN F1 (40 epochs)	SA TA-EN	SA HI-EN	SA ML-EN	SA ES-EN
mBERT	Baseline	62.1	63.4	62.9	69.8	67.3	76.4	60.8
	StdMLM	64.8	65.4	64	74.9	67.7	**76.7**	62.2
	SwitchMLM	**69**	**68.9**	67	-	**68.4**	-	**63.5**
	FreqMLM	68.6	66.7	**67.1**	**77.1**	67.8	76.5	62.5
	StdMLM + ResBERT	66.8_9	64.6_9	64.4_9	77_5	68.4_9	76.6_9	63.1_9
	Sw/FreqMLM + ResBERT	$\mathbf{68.8}_2$	68.9_2	68.1_2	77.4_2	68.9_2	77.1_2	63.7_2
	Sw/FreqMLM + ResBERT + $\mathcal{L}_{aux}$	68_2	$\mathbf{68.9}_2$	$\mathbf{69.8}_2$	$\mathbf{77.6}_2$	$\mathbf{69.1}_2$	$\mathbf{77.2}_2$	$\mathbf{63.7}_2$
XLM-R	Baseline	63.2	63.1	62.7	74.1	69.2	72.5	63.9
	StdMLM	64.4	64.7	66.4	76.0	71.3	**76.5**	64.4
	SwitchMLM	**65.3**	**65.7**	**69.2**	-	**71.7**	-	**64.8**
	FreqMLM	60.8	62.4	63.4	**76.3**	71.6	75.3	64.1

Table 1: QA and SA scores for primary models and language pairs. **Note:** For ResBERT results, the subscript near the F1 scores represents the layer from which the residual connection is drawn for that particular model.

Figure: Results: MLM Pretraining techniques

NN Optimization: Introduction

- Machine learning models aim to learn from data and generalize their knowledge to unseen examples.
- However, as new examples are introduced, the model tends to forget the knowledge it has learned from previous examples, a phenomenon known as "catastrophic forgetting."
- In this work, we propose a novel approach to designing a learning algorithm that takes into account what the model has already learned and adjusts the decision boundary to learn new examples while retaining previous ones.
- We explore the concept of local elasticity in neural networks and how it can be used to design better learning frameworks.

Introduction (Cond.)

- We construct techniques to explicate this locally elastic property of neural networks further — ensuring the decision boundaries are minimally perturbed for every new batch of examples, hence preventing unnecessary forgetting.
- Empirical results show that keeping a buffer of prior examples can help modify the current gradients for faster learning and better generalization.
- A comparative study is also presented in terms of accuracy and time taken for each iteration.

Modifications to SGD

- We worked on possible modifications to the SGD to carefully retain what the model has already learnt and keep learning the new examples
- CustomSGD introduces a simple addition to the standard SGD with Momentum algorithm to make the learning heuristic more aggressive, while maintaining the moving average of gradients.
- We then introduce a class of committee-based SGD, which modifies the gradient we get from model based on gradients from the committee.

CustomSGD

- Model parameters update in SGD:

$$m_t \longleftarrow \alpha m_{t-1} + (1-\alpha) g_t$$
$$w_t \longleftarrow w_t - \eta m_t$$

- We propose the following modifications for CustomSGD :

$$m_t \longleftarrow \alpha m_{t-1} + (1-\alpha) g_t$$
$$k_t \longleftarrow \beta m_t + (1-\beta) g_t$$
$$w_t \longleftarrow w_t - \eta k_t$$

- Where, m_t, g_t, w_t, k_t are the momentum, gradient, model weight and the intermediate momentum term after the t-th iteration respectively.

CustomSGD (Intuition)

- The addition of an intermediate k_t term increases the importance of the gradient of the current batch while keeping the momentum term unperturbed.
- The effect of k_t term is similar to a having a small α in the standard SGD.
- However in this case, the m_t becomes highly susceptible to noisy gradients, which k_t handles very well in the CustomSGD approach.

Committee-based SGD

- We keep a committee of previously seen examples and use them as the representatives to tweak the gradients obtained from the current batch.
- At the start of each training batch, we create a committee and compute the average gradient for each weight.
- Then for each weight, for each example in the batch with gradient g, we compute $g' = \mathcal{G}(g, c)$, where $\mathcal{G}$ is a function that takes in the batch and committee gradients and return a new gradient.
- We use the new gradients instead of the standard batch gradients in the CustomSGD approach explained earlier.

Selecting the committee

- The committee is selected based on two conditions.
- If the number of examples seen so far is less than the committee size, all the previous examples are added to the committee.
- Otherwise, a subset of the previous committee and a subset of current batch is combined to form the new committee.
- The size of the subsets are determined based on the ratio of the batch size and the number of previous examples seen.
- This approach ensures that the committee is updated with the new data while retaining some of the previous knowledge

Committee-based SGD approaches

- We get multiple variants of committee-based SGD with the underlying $\mathcal{G}(\cdot, \cdot)$ we use:
  - CommitteeSGD
  - CommitteeSGD2
  - CommitteeSGD2Positive
  - StaleCommitteeSGD
  - StaleCommitteeSGD2

CommitteeSGD

- We define the g' as:

$$g' = \hat{g}(|g| - f(\langle g, c \rangle))$$

- Where, $\hat{g}$ is the unit vector in the direction of g, $|g|$ its magnitude. f is monotonically decreasing function and c is the average gradient from the committee.
- The intuition behind is to retain the existing weights if the gradient direction overlaps with the accumulated committee and updates the weights according to the current batch if the committee and batch have different expected updates.

CommitteeSGD (Cond.)

- f is selected to be a monotonically decreasing function as to reduce the magnitude of the gradient when the inner product between the current gradient and the committee gradient is negative, and leave it unchanged when the inner product is positive.
- This is achieved by using a non-linear function, such as Decreasing ReLU or Decreasing Leaky ReLU, which is applied to the gradient before it is used to update the model parameters:
  - $DReLU : f(x) = -x$ for $x < 0$, 0 otherwise
  - $DLReLU : f(x) = -x$ for $x < 0$, $f(x) = \alpha * -x$ otherwise

CommitteeSGD2

- We define the g' as:

$$g' = g - \hat{c}f(\langle c, \hat{g} \rangle)$$

- This heuristic variant helps us understand how much the committee's previous examples influence the current batch.
- We calculate the magnitude to be subtracted from the gradient based on the value of $f(\langle c, \hat{g} \rangle)$, which indicates the committee's impact on the current batch.
- This helps us update the model in a way that reduces the changes in the direction that might have affected the predictions of the committee.

CommitteeSGD2Positive

- We define the g' as:

$$g' = g - \hat{c}\max\left(\langle g, \hat{c}\rangle, f(\langle c, \hat{g}\rangle)\right)$$

- The update is similar to the CommitteeSGD2 , we just have the additional constraint of max perturbation possible to keep the gradients we want to modify, positive
- This improves the robustness to noise and keeps a check on the max perturbation we are allowing by our committee-based approaches.

Stale Committee SGD approaches

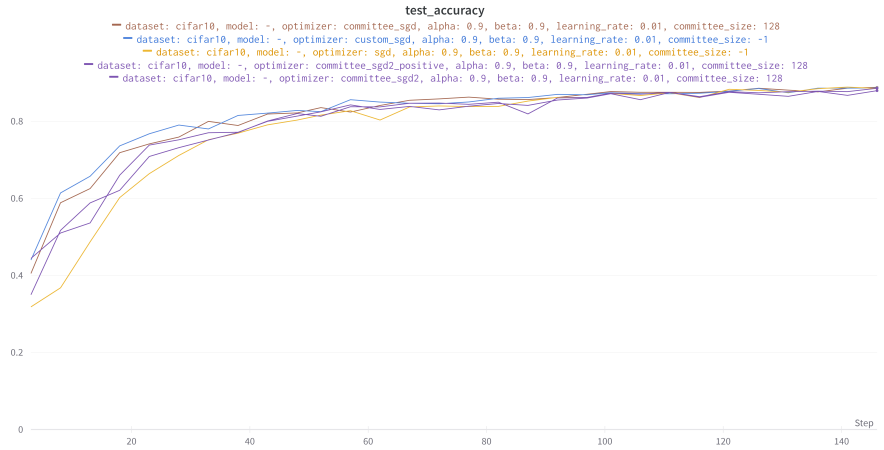
- The StaleCommitteeSGD and StaleCommitteeSGD2 approach, we keep the update equation same as CommitteeSGD and CommitteeSGD2 respectively, but we do not update the gradient of the committee for a fixed set of batches.
- This reduces the overhead we incur with the earlier approaches, where we have an extra pass of gradient computation on the committee at every iteration.
- Stale versions use the old gradients for a few iterations before recalculating/refreshing the gradients. This amortizes the overall cost of the optimizations.

Evaluation tasks for our approaches

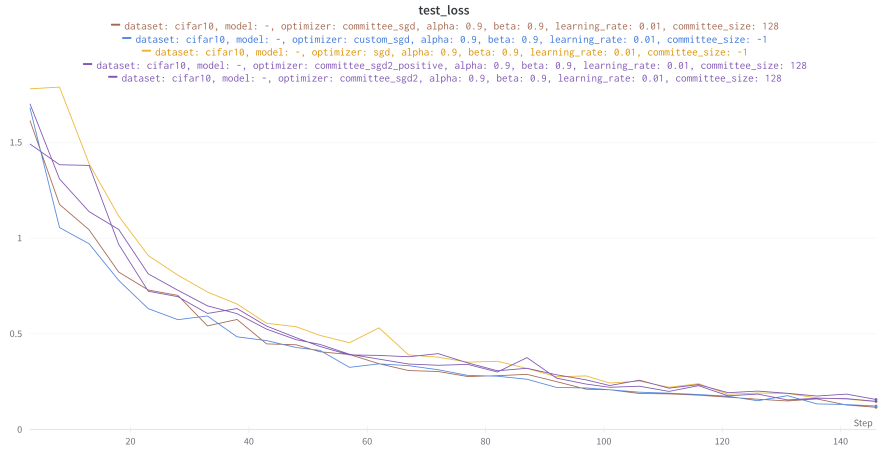
- We evaluate our SGD modifications on the following tasks:
  - **CIFAR-10**: A computer vision task where the goal is to classify 60,000 32x32 color images into 10 classes.
  - **CIFAR-100**: A computer vision task where the goal is to classify 60,000 32x32 color images into 100 fine-grained classes.
  - **WMT16 (CS-EN)**: A machine translation task where the goal is to translate a sentence from English to Czech or vice versa.
  - **IMDB movie reviews**: A sentiment analysis task where the goal is to predict whether a movie review is positive or negative.
  - **FLower-102**: Flower 102 task involves classification of images of flowers into 102 different categories.

Results: CIFAR-10 (VGG16)

- CustomSGD outperforms standard SGD with Momentum significantly
- Committee-based methods do not provide additional gain over CustomSGD
- **Reason:** Fewer classes (10) means less diversity in learned representations, reducing benefit from committee mechanism



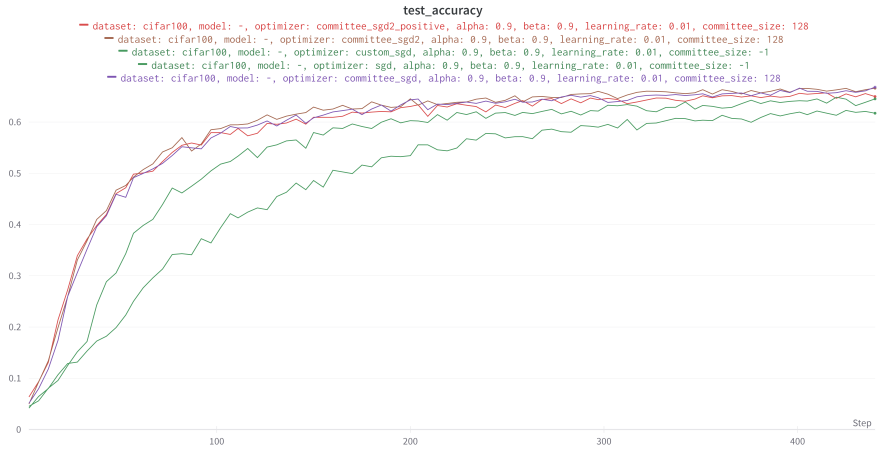
(a) Test accuracy



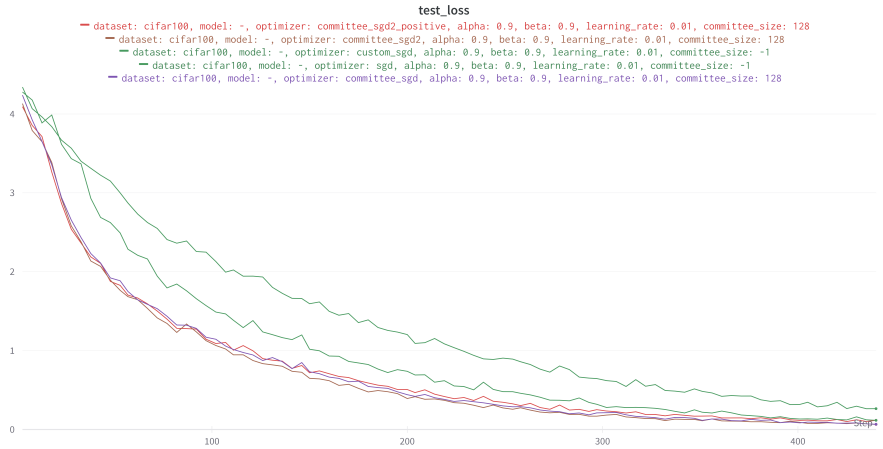
(b) Test loss

Results: CIFAR-100 (VGG16)

- Committee-based methods show clear benefits on CIFAR-100
- **Key finding:** Refreshing committee improves accuracy significantly
- Performance ordering: CommitteeSGD (refresh) $\gg$ CustomSGD $\approx$ CommitteeSGD $>$ SGD
- Accuracy: **0.6682** (refresh) vs **0.6335** (standard)



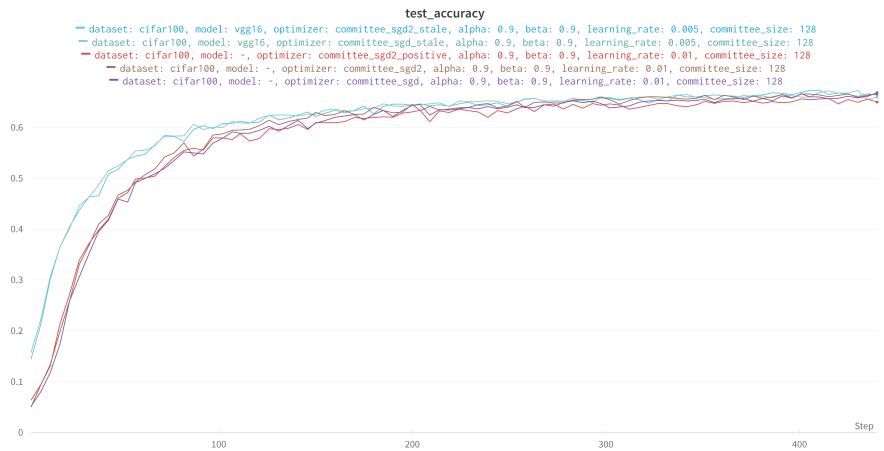
(a) Test accuracy



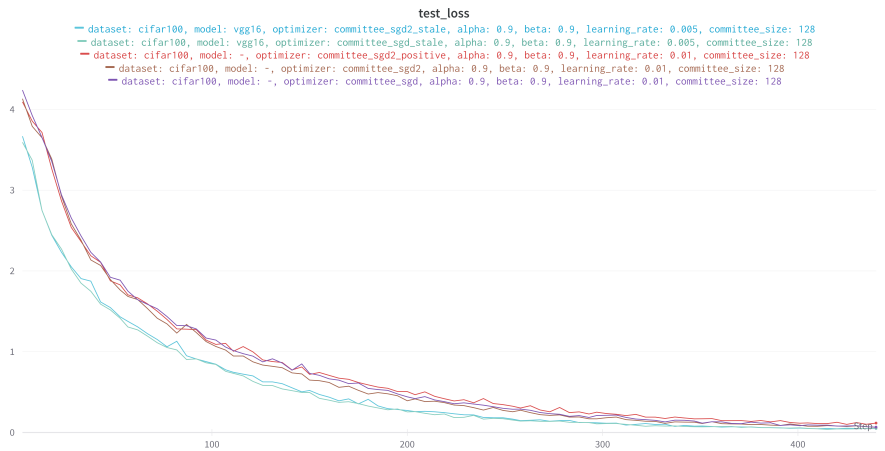
(b) Test loss

Results: CIFAR-100 (Stale Committees)

- Stale variants maintain competitive performance
- Refresh rate of 5 batches achieves similar accuracy to non-stale version
- **Benefit:** Significantly reduced computational overhead



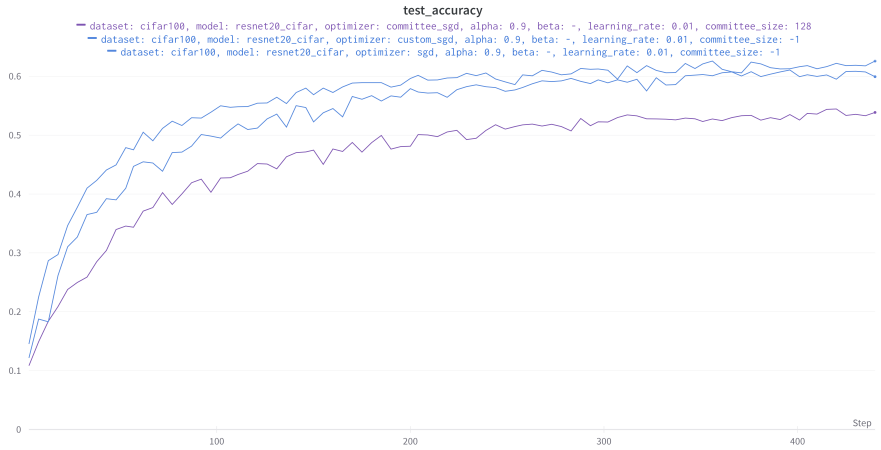
(a) Test accuracy



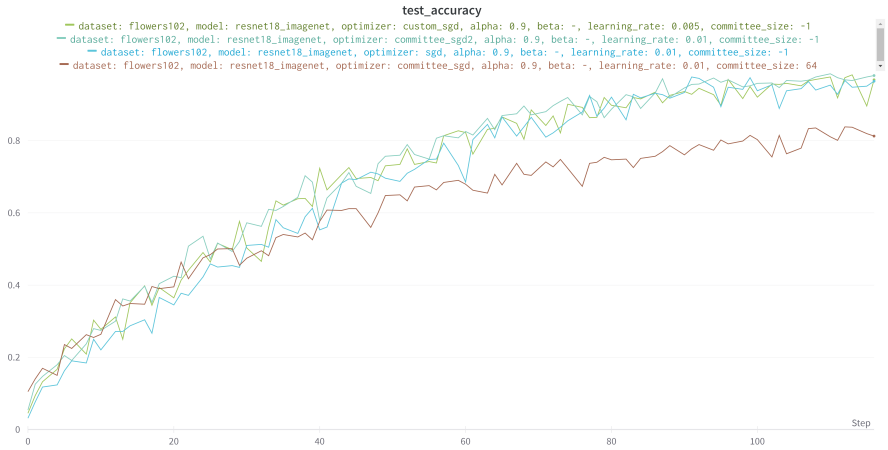
(b) Test loss

Results: ResNet Architectures

- Our SGD modifications show **limited improvement** on ResNet models
- Observed on both CIFAR-10 (ResNet20) and Flowers102 (ResNet18)
- **Hypothesis:** Skip connections in ResNet provide inherent protection against catastrophic forgetting
- Committee-based approach most effective for VGG-style architectures



(a) CIFAR-100 ResNet20

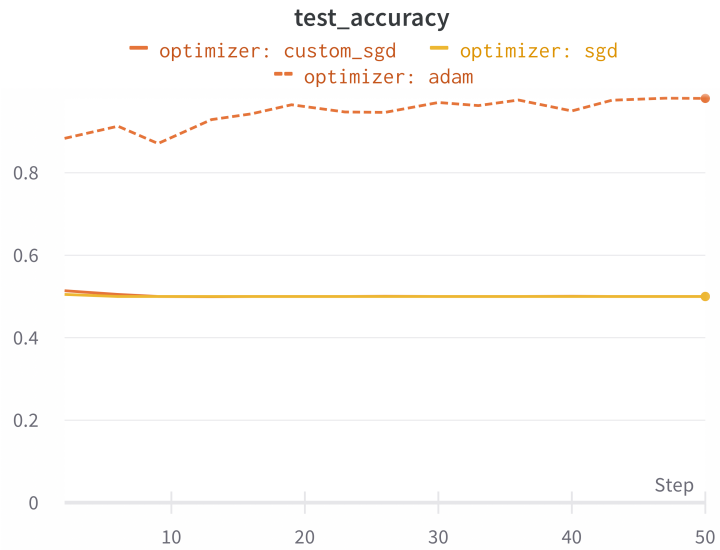


(b) Flowers102 ResNet18

Results: NLP Tasks

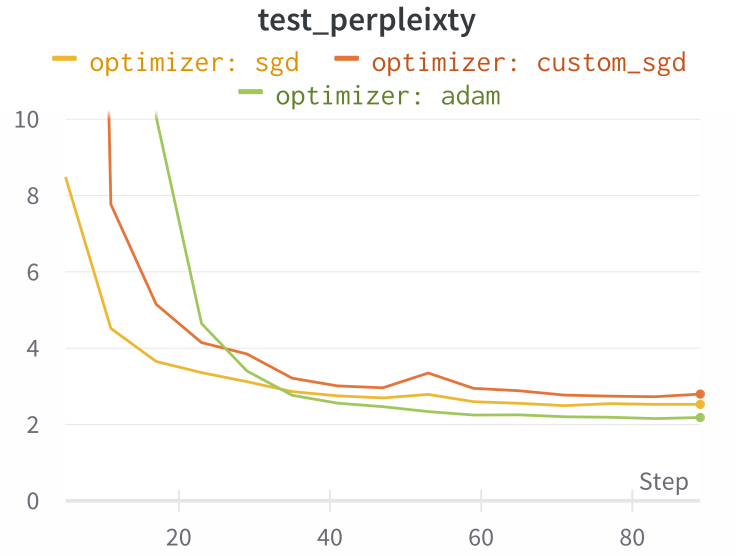
IMDB Sentiment Analysis

- CustomSGD improves convergence
- Committee variants show modest gains
- Methods generalize beyond vision



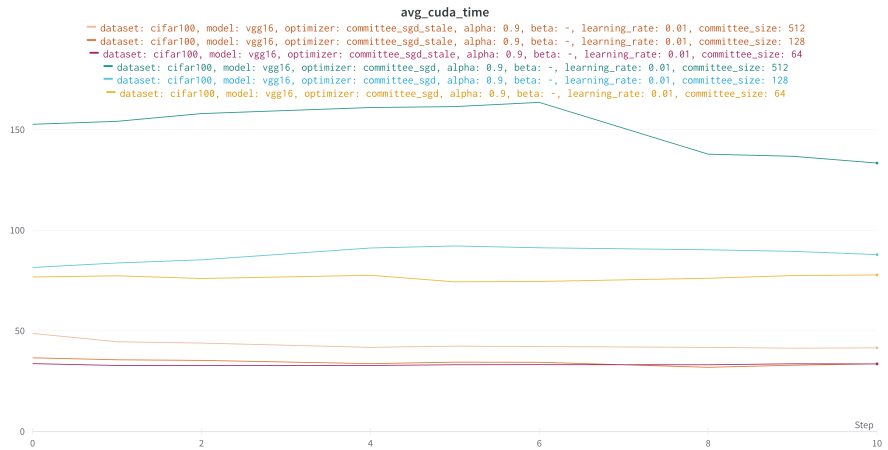
WMT-16 Translation

- Competitive perplexity scores
- Benefits for seq2seq models
- Retains translation patterns

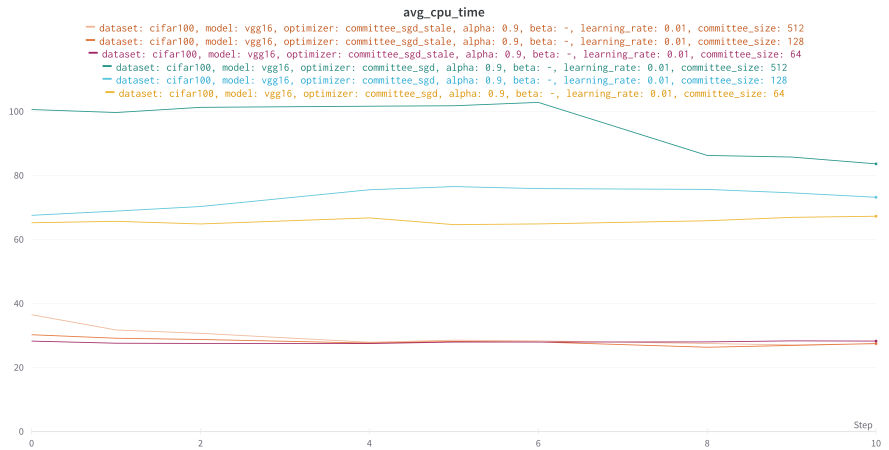


Profiling Results

- Committee-based SGDs incur overhead from extra backward pass
- **Stale variants** significantly reduce this overhead
- With refresh rate of 5 batches: near-parity with CustomSGD in wall-clock time



(a) Average CUDA time



(b) Average CPU time

Results Summary

Method	CIFAR-10	CIFAR-100
SGD + Momentum	Baseline	Baseline
CustomSGD	↑ Significant	↑ Moderate
CommitteeSGD	≈ CustomSGD	↑ Moderate
CommitteeSGD (refresh)	≈ CustomSGD	↑↑ Best
StaleCommitteeSGD	≈ CustomSGD	↑ Good + Fast

Key Takeaways:

- More classes → more benefit from committee-based methods
- VGG architectures benefit more than ResNets
- Stale variants offer best cost-accuracy trade-off

Conclusion

Our main contributions can be summarised as:

- Modifying standard SGD by adding an intermediate momentum term to handle noisy gradients and improve over all training performance.
- Introducing committee based SGD approaches which tweaks the gradients calculated from the current batch based on the committee gradients, which addresses the catastrophic forgetting.

Future Work

Future work:

- Currently we select committee members randomly, we plan to work out a more appropriate selection strategy which focuses on identifying relevant members from the previously seen examples.
- SGD optimization is not relevant for NLP, Audio Process, and other larger tasks. We plan to extend this idea of Custom and Committee based SGD to ADAM optimizer.
- Stale Committee SGDs address the computational overhead but fails to achieve desired performance, we need to find a solution which achieves both the target.