Accelerating NLP Model Training and Enabling Higher Accuracy for Financial Services Applications

The benefits of training from scratch using domain-specific datasets can now be realized in an enterprise environment thanks to Cerebras AI accelerator technology.

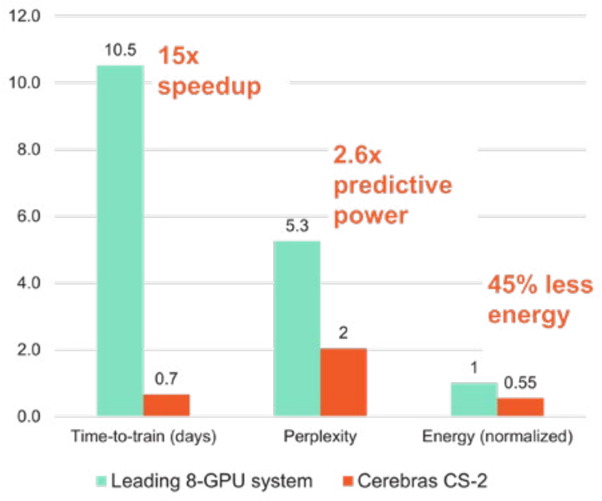
Sanjana Mallya, Cindy Orozco Bohorquez and Natalia Vassilieva, Cerebras Systems

Summary

This report shows the results of a pilot project conducted by a major financial services institution and Cerebras Systems. The project aimed to improve $\text{BERT}_{\text{LARGE}}$ model accuracy for financial services applications by training the model from scratch using domain-specific data, rather than simply using a generic model trained on general purpose text as a starting point

The project also demonstrated that training from scratch, which was previously intractable using conventional hardware, could be made easy and far faster using the Cerebras CS-2 system. This performance and ease of use made rapid experimentation with model parameters, and thus a better-performing model, practical in an enterprise environment.

The CS-2 system reduced training time by 15X compared to a leading 8-GPU server, delivering the compute performance of more than 120 GPUs. This impressive speedup enabled us to demonstrate dramatic improvements in resulting model quality using the Thomson Reuters TRC2 dataset, while almost halving energy consumption.



Contents

Introduction

Large neural language models, such as BERT, have demonstrated impressive results on many natural language processing (NLP) tasks. A common approach is to take a model that has been pre-trained using general domain corpora, such as text extracted from web pages, then fine-tune the generic model using a domain-specific training set. However, past work has shown that for specialized domains with abundant domain-specific texts, pretraining language models from scratch using domain-specific data results in more accurate models[i]. Despite this evidence, large language models are rarely pre-trained from scratch on domain-specific data, because this pre-training process is very computationally intensive. The CS-2 system significantly reduces time and cost of large language model pre-training and makes it easier to experiment with domain-specific language models.

During this project, we employed a single CS-2 system to train and compare several BERT model configurations. The model, dataset and vocabulary were chosen to match the customer's interests. The focus of this POC is on the unsupervised pre-training of BERT to learn meaningful representations of financial corpus of data that can be used for a myriad of downstream fine-tuning tasks. We measured the quality of the trained models with standard evaluation metrics for BERT: masked language model accuracy, next sentence prediction accuracy and masked language model perplexity. Perplexity can be thought of as the measure of uncertainty – how uncertain a model is when predicting the next word in a sentence, or as a measure of prediction error. Low perplexity score indicates that the model is not "perplexed" by the generated text and will accurately predict the next word. A $\text{BERT}_{\text{LARGE}}$ model previously trained on a general domain corpus (Open Web Text) served as a baseline. A $\text{BERT}_{\text{LARGE}}$ model trained from scratch on the Reuters TRC2 dataset[ii] demonstrated the best result, with perplexity of more than 2x lower compared to the baseline.

To compare wall-clock training time with a CS-2 system, the customer replicated training of $\text{BERT}_{\text{LARGE}}$ model on the Reuters TRC2 dataset on a leading 8-GPU server. The preliminary performance data, reported by the customer, indicates that a single CS-2 system completes end-to-end training 15x faster than the 8-GPU server. Because of the sub-linearity of GPU performance scaling for such tasks, this result implies that the CS-2 delivers the compute performance of far more than 15 x 8 = 120 GPUs for this work.

BERT

Bidirectional Encoder Representations from Transformers or BERT[iii], is a leading language model architecture that is used for a wide range of NLP applications. We employed a Cerebras CS-2 system to train BERT to learn a domain specific language model by training on the Reuters Financial dataset tokenized with FinVocab[iv], a domain specific vocabulary. BERT has a modular architecture comprised of input embeddings, a stack of encoder blocks, and task-specific output heads. The model can be scaled to larger parameter counts and model capacities by increasing the number of encoder blocks (L), number of self-attention heads (A), and hidden size (H) in the model architecture.

For this POC, we have experimented with various flavors of BERT model. We first obtained results for a smaller BERT model with 8 encoder blocks and hidden size of 1024 (L=8, H=1024, A=16) and then with BERT Large 24 with encoder blocks and the same hidden size of 1024 (L=24, H=1024, A=16).

Datasets

The dataset we primarily used for our experiments is the Thomson Reuters Text Research Collection (TRC2), a well-known public dataset for training language models used in the financial domain. The TRC2 corpus comprises 1,800,370 financial news stories covering the period from January 1, 2008, to February 2, 2009.

The dataset was cleaned to remove tags, date/time stamps, bad characters, and other extraneous information. The overall size of the data after cleaning is 2.1GB.

We converted the data into the TensorFlow TFRecord custom data format which is a simple format for storing a sequence of binary records. TFRecord files are natively integrated into TensorFlow's tf.data API, enabling easy batching, shuffling, and caching. To facilitate easy creation of TFRecords, we split the dataset into 29,047 smaller files. To create the training and validation set, we used 28,747 files for the train set and 300 files for the validation set. We created two datasets from the same data, differing by Maximum Sequence Length (MSL). For each sentence/sequence in the dataset, we define a limiting number of tokens that comprise the sequence beyond which tokens are added to the next training sample. The sequences are also shuffled. One dataset comprises of TFRecords with MSL 128 and the other with MSL 512. The size of the training set is 37GB, and the size of the validation set is 349MB. During training, we follow a two-phase approach which is standard for BERT training. We first train on shorter sequence MSL 128 dataset for a few hundred thousand steps and further train on longer sequence MSL 512 dataset.

We used the OpenWebText (OWT)[v] dataset for non-domain-specific training. OWT is an open-source version of the WebText corpus, an internal OpenAI corpus created mainly for training OpenAI's GPT-2 model[vi]. The text is web content extracted from URLs shared on Reddit with at least three upvotes. The size of the train dataset for MSL 128 case is 49GB and for the MSL 512 case is 56GB.

Choice of vocabulary

In conjunction with the customer, we chose a domain-specific vocabulary FinVocab. Based on the results in the paper[iv], we expected this vocabulary to yield better results on several downstream tasks. FinVocab was contrived for the training of FinBERT. It is a Word Piece Vocabulary derived out of the financial data used for FinBERT using the SentencePiece library. We used the uncased version of the Vocabulary which has a total of 30,873 tokens. This is very similar to the 28,996 and 30,522 token sizes of the original BERT uncased BaseVocab. The resulting overlap between the original BERT BaseVocab, and FinVocab is 41% for both the cased and uncased versions.

Effect of tokenizing with a financial vocabulary

Figure 2 shows the top 30 most frequently occurring tokens in the tokenized TRC2 dataset. The first plot shows the most popular tokens when a specialized financial vocabulary is used, such as FinVocab. The second plot shows the most popular tokens when we tokenize with generic BERT Vocab, that is a general-use vocabulary. First, regardless of the vocabulary used for tokenization, we encounter some commonly occurring general words and special tokens. More importantly, by using FinVocab, we see that about 50% of the top 30 tokens are financial relevant terms, vs. under 10% using generic BERT Vocab. In Figure 3 we see the effect of tokenizing the OWT dataset with FinVocab. We observe that at least 30 % of the tokens are financial relevant terms.

Figure 4 is a histogram plot of the 30 least occurring tokens in the tokenized dataset as per FinVocab. As seen from the plot, when FinVocab is used most of the tokens are unique financial vocable, alphanumeric, hyphenated or with attached punctuation mark. With the generic BERT vocabulary only a small fraction is financial related terms. In Figure 5 we again saw a larger fraction of the tokens belonging to FinVocab.

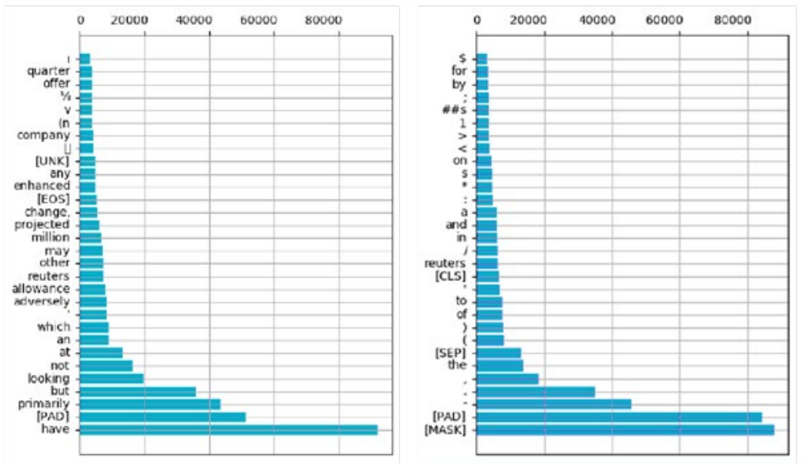


Figure 2. Top 30 tokens by frequency in TRC2 dataset using FinVocab (left) and generic BERT Vocab (right).

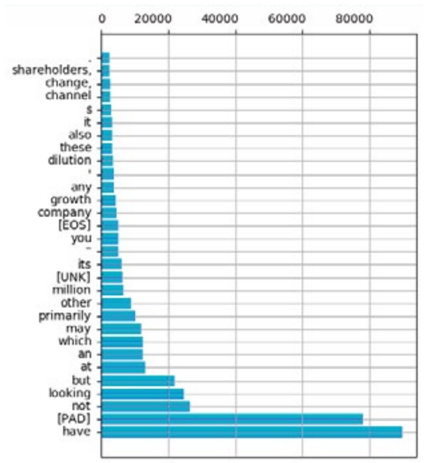


Figure 3. Top 30 tokens in OWT dataset tokenized using FinVocab

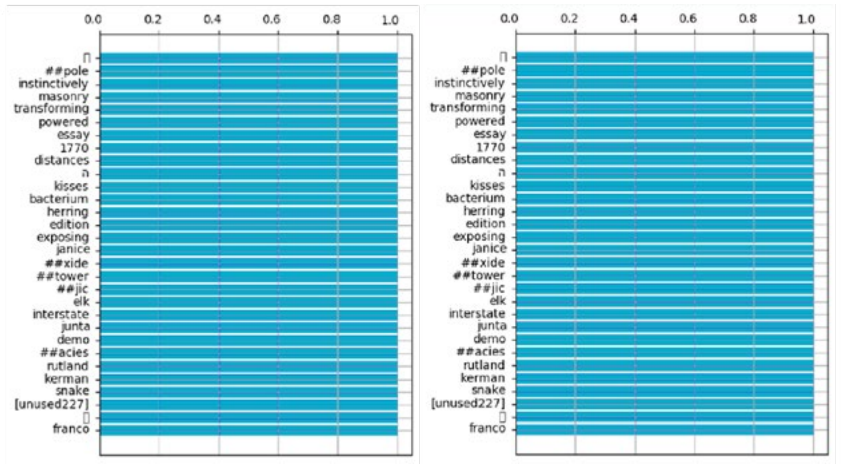


Figure 4. 30 Least occurring tokens by frequency in TRC2 dataset using FinVocab (left) and generic BERT Vocab (right), each of which occurs only once.

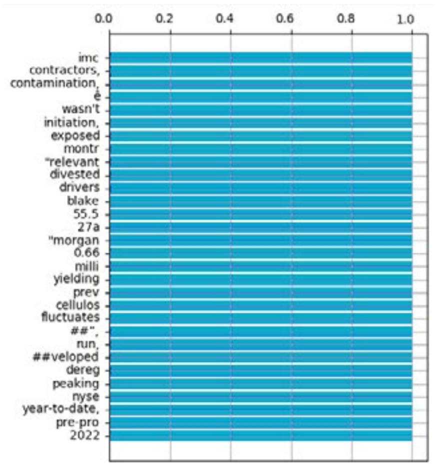


Figure 5. 30 Least occurring tokens in OWT dataset tokenized using FinVocab.

Suitability of FinVocab for TRC2 and OWT datasets
We also analyzed some statistics after parsing 64,000 random samples from different datasets. We considered two datasets, TRC2 and OWT, each tokenized using FinVocab for a Maximum Sequence Length of 128 and 512. Our goal was to see that FinVocab generates the same number of tokens in a sample of both datasets, and also to see if FinVocab does a better job identifying known tokens in TRC2 data than in OWT data. Additionally, we also observed how TRC2 tokenized using generic BERT Vocab appears.

We considered the total tokens seen, the numbers of tokens that were split up, the average MSL across all the sentences in each of the examples and the total number of <UNK> tokens. <UNK> tokens are used when a word (or a sub word from a word generated by the tokenization algorithm used) in the dataset does not have match in the vocabulary.

Data-set	Vocab	MSL	Num of Examples	Total Tokens	Number of Splits	Ave. MSL	Total Unknown Tokens	<unk> as a % of total tokens
TRC2	Fin	128	64,000	7,725,720	449,833	120	45,715	0.59%
		512	64,000	27,900,860	1,651,748	435	171,425	0.61%
OWT	Fin	128	64,000	7,501,159	510,529	117	60,610	0.80%
		512	64,000	22,828,450	1,599,624	357	185,206	0.81%
TRC2	Ge-neric	128	64,000	7,363,510	694,496	115	26	0.00035%
		512	64,000	25,650,617	2,429,914	401	54	0.0002%

Table 1. Statistics of some metrics of a randomly selected TF record sample of TRC2 and OWT dataset tokenized using FinVocab and TRC2 dataset tokenized using generic BERT Vocab.

From Table 1 we can notice:

Same number of total and masked tokens. Comparing the MSL 128 versions of the TRC2 and OWT datasets, as well as the MSL 512 versions of the dataset, we see that they roughly have the same number of total tokens as well as masked tokens.

Significant difference between average sequence length and maximum sequence length.
Each sample has a different sequence length that is upper bounded by the maximum sequence

length (MSL), being either 128 or 512 tokens. For sequences with a shorter sequence length than the MSL, we traditionally used padding tokens to have a sample of sequence length equals to MSL. As we can see, the average sequence length for both TRC2 and OWT is well below the Max Sequence Length (128 and 512). This becomes an advantage when training with variable sequence length (VSL). VSL is a special feature of our software that allows padding tokens to be skipped, and only meaningful tokens are processed. This feature provides a significant performance boost.

FinVocab better fits TRC2. The <UNK> tokens constitute a lesser percentage of the total tokens for TRC2 than for OWT (for both the MSL 128 and MSL 512 versions), indicating that FinVocab is a better fit for the TRC2 data.

More Splits with OWTVocab. We observe a very low percentage of <UNK> tokens as a percentage of all the tokens for TRC2 data tokenized using generic BERT Vocab. But we also observe that a much larger number of tokens were split. This is explainable by the fact that generic BERT Vocab has a wider variety of general terms and fits the dataset by splitting words (like financial terms) into smaller words that have matches in the Vocabulary. This inhibits the model's learning capacity in the financial domain.

Pre-training protocol

The pre-training followed the standard BERT pre-training protocol with the Adam[vii] optimizer. The learning rate was linearly increased (warmed up) from zero to the peak learning rate in the first 10000 steps, then underwent a linear decay to zero for the remaining of training. The peak learning rate for all runs and models is 0.001. Whenever we switch datasets or MSL for the same model, the learning rate is picked up from the last step.

Tag	BERT Config	Train Dataset and Steps			Batch Size	Variable Seq. Length
8-Encoder	BERT 8 Encoder	TRC2 MSL 128 (500k Steps)			256	Disabled
				TRC2 MSL 512 (200k Steps)	256	Disabled
Large-O-T	BERT Large	OWT MSL 128 (400k Steps)	TRC2 MSL 128 (500k Steps)		256	Enabled
				TRC2 MSL 512 (100k Steps)	256	Enabled
Large-T	BERT Large	TRC2 MSL 128 (900k Steps)			256	Enabled
				TRC2 MSL 512 (100k Steps)	256	Enabled
Large-T-1k	BERT Large	TRC2 MSL 128 (225k Steps)			1000	Enabled
				TRC2 MSL 512 (25k Steps)	1000	Enabled
Large-O	BERT Large	OWT MSL 128 (900k Steps)			256	Enabled
				OWT MSL 512 (100k Steps)	256	Enabled

Table 2 A summary of the different models with their configurations. Other model parameters such as learning rate, decay policy remain the same across models.

The standard approach for BERT pre-training introduced in the original BERT paper[iii] usually follows a two-phase approach: training with MSL 128 for 90% of the total steps and MSL 512 for 10% of the total steps. This was found to accelerate training while still achieving higher accuracies for longer sequences. We also mostly followed this protocol except as listed in the table above.

To observe convergence and performance behavior for the corpora tokenized with a domain specific vocabulary, we began our experiments with a small BERT model, 8-Encoder-T. After observing acceptable values for training loss and seeing a consistent loss curve, we moved on to $\text{BERT}_{\text{LARGE}}$ models. In our experience, for pretraining on different corpora, $\text{BERT}_{\text{LARGE}}$ models have demonstrated good accuracy scores.

With $\text{BERT}_{\text{LARGE}}$ models, we experiment with the effect of pretraining from scratch on domain specific data (TRC2), pretraining from scratch on a slightly different dataset before training on TRC2 and pretraining with a larger batch size on TRC2. The main aim of these experiments, in line with goals defined in the introduction is to analyze and assess the effect of pretraining from scratch on accuracy and the affect different parameters have on the speed of training.

As mentioned above, variable sequence length (VSL) a special feature of Cerebras software that allows padding tokens to be skipped, and only meaningful tokens are processed. This feature provides a significant performance boost. The buckets used for VSL-based training for each of the models are shown in Table 3.

Tag	VSL Buckets Used
Large-O-T	OWT MSL 128: [36, 47, 57, 66, 76, 85, 95, 106, 117] TRC2 MSL 128: [33, 44, 54, 63, 73, 83, 94, 105, 116] TRC2 MSL 512: [60, 86, 117, 154, 198, 245, 301, 366, 436]
Large-T	TRC2 MSL 128: [33, 44, 54, 63, 73, 83, 94, 105, 116] TRC2 MSL 512: [60, 86, 117, 154, 198, 245, 301, 366, 436]
Large-T-1k	TRC2 MSL 128: [33, 44, 54, 63, 73, 83, 94, 105, 116] TRC2 MSL 512: [60, 86, 117, 154, 198, 245, 301, 366, 436]
Large-O	OWT MSL 128: [39, 56, 72, 89, 108] OWT MSL 512: [86, 155, 237, 324, 415]

Table 3. Buckets used for variable sequence length (VSL) training

Pre-training evaluation

The sum of the MLM and NSP task losses are used to track the progress of the pretraining. MLM loss is calculated as the Cross Entropy loss between the predicted logits of the input tokens that were masked (about 15 % inputs tokens are masked) and their labels. Next Sentence Prediction (NSP) Loss is a Binary Cross entropy loss where the embeddings of the [CLS] and the [SEP] tokens are used to predict whether two sentences appearing together are coherent. We also get MLM and NSP accuracy scores on the held out eval dataset for intermittent checkpoints to assess the quality of training. The held out eval dataset for all tasks is the TRC2 Eval dataset curated as a subset of the whole dataset, as discussed in the Datasets section of this report.

Figure 6 shows a plot of the training loss as a function of the number of steps for all the BERT models. The figure also contains eval loss obtained from checkpoints during the two stages of training. As can be observed from the plot, the Large-T model converges quickly and maintains the lowest training loss. Large-O-T model catches up quickly with the BERT-Large-T model. Another trend that can be observed from the plot is that the eval loss from MSL-512 checkpoints for all models other than Large-T-1k are lower than the respective model's MSL128 checkpoint.

Figure 7 is also a plot of training loss, but as a function of samples seen. This plot is better suited to observe a model's learning after seeing the same amount of data. Here we can observe that Large-T-1k MSL128 training phase has a loss comparable to that of the Large-T model. Other trends observed are the same as those noted from Figure 6.

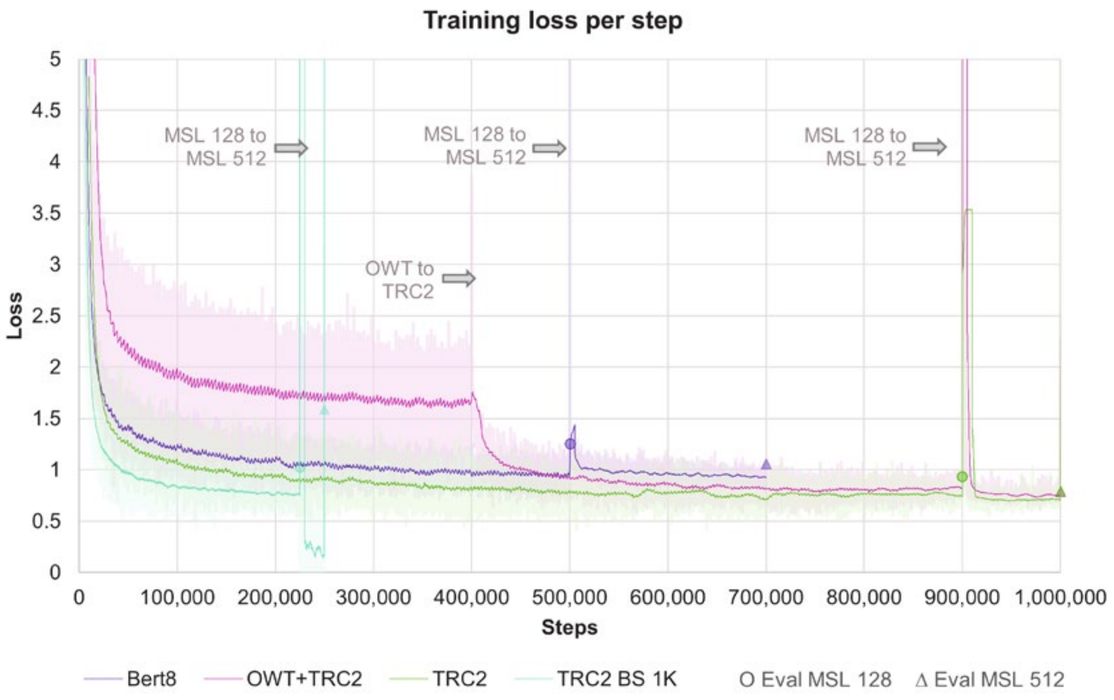


Figure 6. Loss curves for 8-Encoder-T and BERT$_{LARGE}$ Large pre-training on CS-2. The large swings in loss are due to the change in MSL from 128 to 512. The different number of steps is due to different batch sizes used.

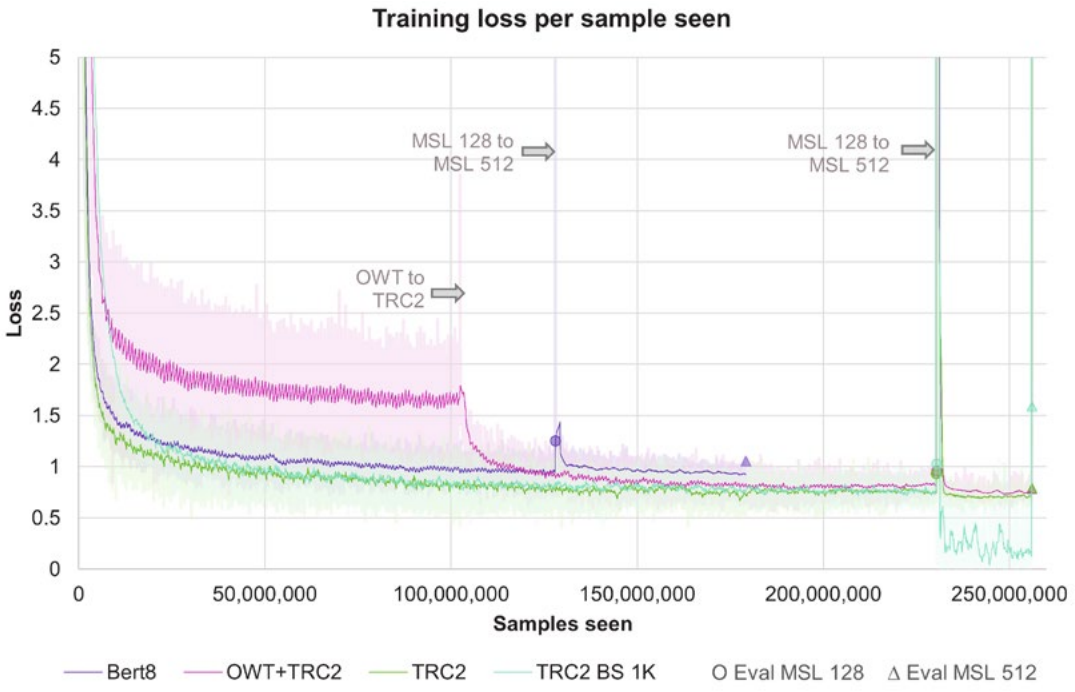


Figure 7. Loss curves for 8-Encoder-T and BERT$_{LARGE}$ pre-training on CS-2 with respect to samples seen. This plot better reflects the learning regimes of the models with different batch sizes.

Tag	Eval	MLM Accuracy	NSP Accuracy	Perplexity	Eval Loss
8-Encoder-T	TRC2 128 Val	0.783	0.968	2.922	1.254
	TRC2 512 Val	0.826	0.977	2.191	1.058
Large-O-T	TRC2 128 Val	0.805	0.974	2.262	0.928
	TRC2 512 Val	0.829	0.984	2.058	0.775
Large-T	TRC2 128 Val	0.814	0.976	2.238	0.951
	TRC2 512 Val	0.837	0.984	2.044	0.786
Large-T-1k	TRC2 128 Val	0.797	0.972	2.452	1.027
	TRC2 512 Val	0.721	0.973	4.428	1.569
Large-O					
	TRC2 512 Val	0.690	0.770	5.254	3.000

Table 4. Accuracy, perplexity, and Eval Loss of pre-trained BERT models on MLM and NSP tasks using a validation dataset excluded from training.

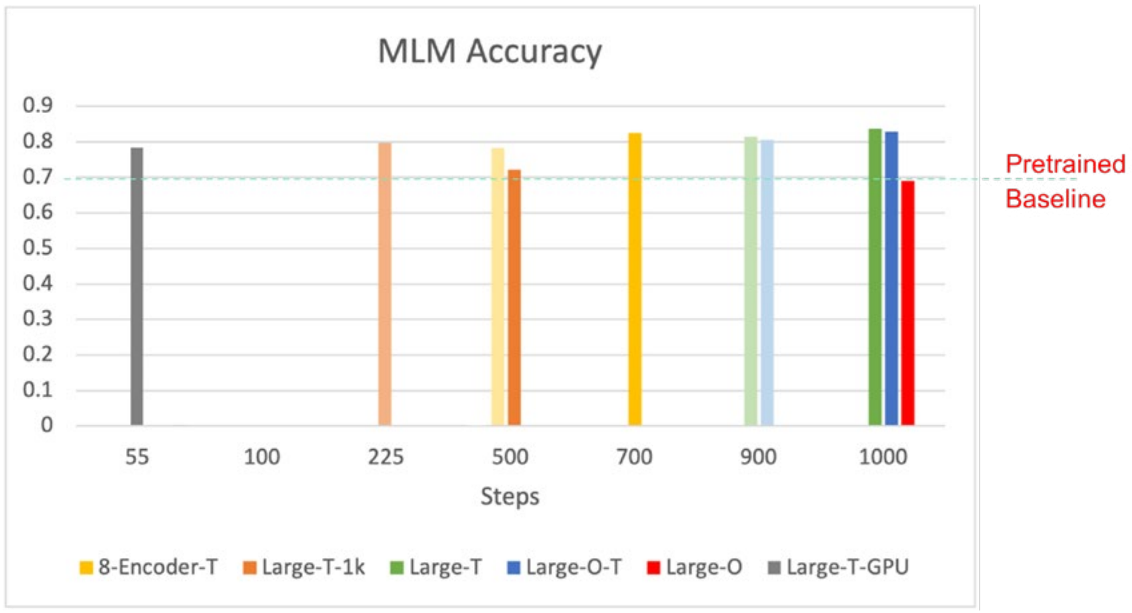


Figure 8. A comparison of MLM accuracy across pre-trained BERT models

Figure 8 shows that the MLM accuracy for models 8-Encoder-T-T, Large-O-T, Large-T increased with further pre-training on MSL 512 dataset. We also observed that the pretrained baseline, Large-O has the lowest MLM accuracy.

We observe from Figure 9 that for all models, with further pre-training on MSL 512 dataset, there is a slight increase in NSP accuracy. Again, the pretrained baseline, Large-O has the lowest NSP accuracy.

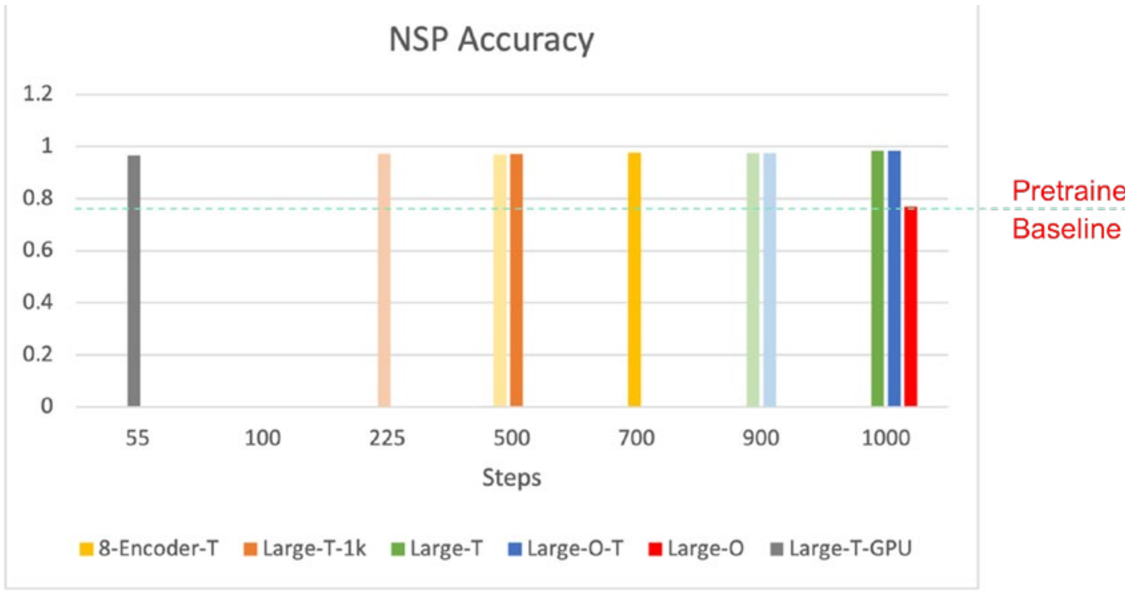


Figure 9. A comparison of NSP accuracy across pre trained BERT models

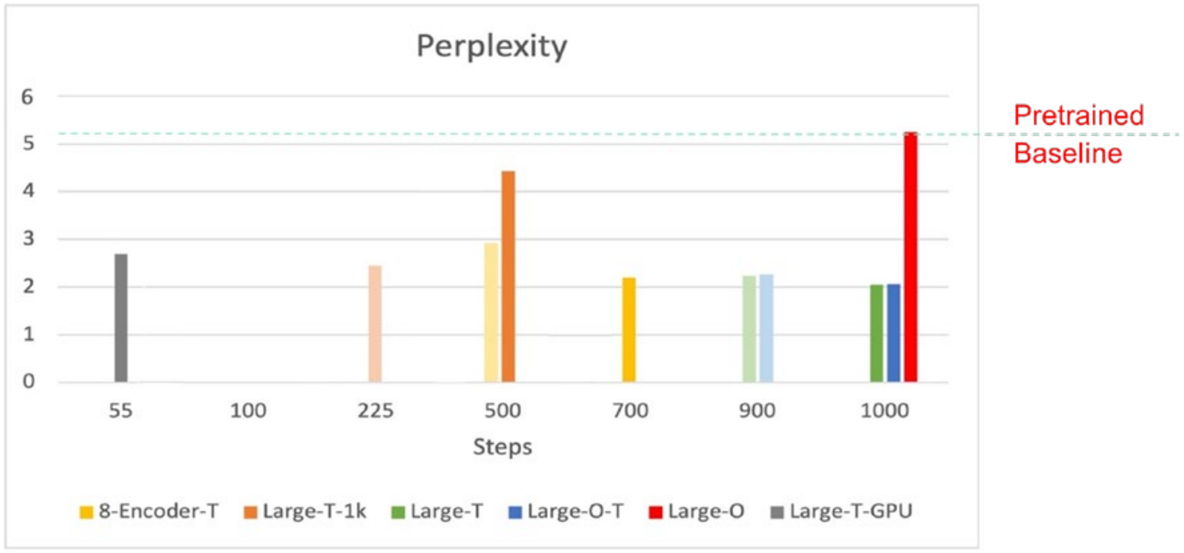


Figure 10. A comparison of Perplexity scores across pre-trained BERT models

We observe from the Figure 10 that the perplexity for all models 8-Encoder-T, Large-O-T, Large-T, decreases with further pre-training on MSL 512 dataset. The smaller 8-Encoder-T model sees the most significant decrease in accuracy. We observe the highest perplexity with the baseline Large-O model.

To summarize:

- Large-T saw the best NSP and MLM accuracy highlights goes to show the benefit of pre-training from scratch on domain specific data.
- Large-O-T saw the lowest eval loss. shows that pretraining on non-domain specific data before training on domain specific data does improve generalization. The accuracy scores and perplexity of this model is also very close to that of Large-T.

Pre-training performance

We recorded two pre-training performance metrics: The total time taken, and the power consumed to train the model to a certain perplexity (Table 5, Figure 11, Table 6, Table 7, Table 8). For this comparison, the customer chose to run the same code on their GPU system that we developed for the CS-2 system. We do not claim that this is the best achievable performance on this type of model. It is likely that further optimization would improve the performance of both configurations.

In Table 5, we record the total training time for all the models, where each model has parsed the same number of examples. Some observations from the table are:

- As expected, a smaller model (8 Encoder) takes less time to train for the same number of examples.
- Large-O-T and Large-T approximately see the same training time as they have the same model architecture but see different data.
- Large-T-1k takes lesser time than other large models as we are able to utilize the CS-2 fabric better with a larger batch size.

Tag	Batch Size	Samples/sec		Training time (hours)		
		MSL 128	MSL 512	MSL 128	MSL 512	Total
8-Encoder-T	256	8,346	2,097	4.3	6.8	11.0
Large-O-T	256	6,016	1,280	10.6	5.6	16.2
Large-T	256	5,888	1,280	10.9	5.6	16.4
Large-T-1k	1,024	6,840	1,536	9.4	4.6	14.0

Table 5. Throughput and training times for each model and MSL.

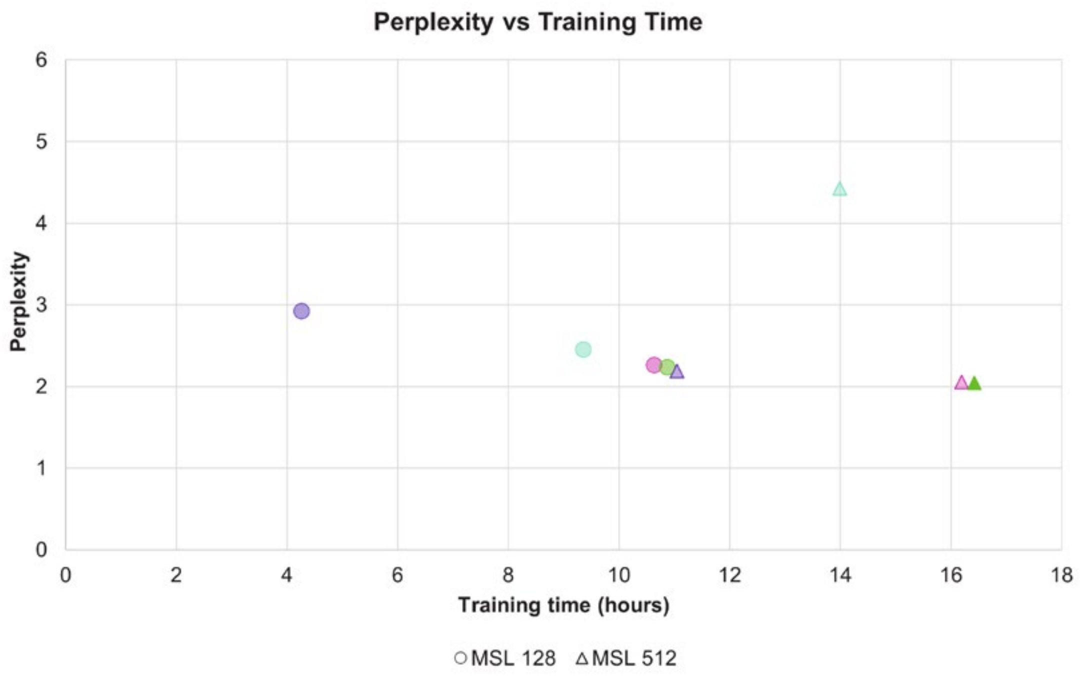


Figure 11. Total training time vs perplexity for different models and MSLs. The labels follow the same color code as in Table 2.

MSL	Number of steps	Device	Steps/sec	Time (hours)
128	100,000	CS-2	23.00	1.21
		8-GPU	2.08	13.35
512	55,000	CS-2	5.00	3.05
		8-GPU	0.42	36.37

Table 6. Training time for Large-T model on CS-2 vs 8-GPU system for recorded steps from customer's GPU checkpoint.

Setup	MSL 128				MSL 512				Total time
	Batch	Steps/s	Required steps	Required time	Batch	Steps/s	Required steps	Required time	
CS-2	256	23.00	900,000	10.9 h	256	5.00	100,000	5.6 h	16.5 h
8-GPU	256	2.08	900,000	120.2 h	128	0.42	200,000	132.28 h	252.5 h 10.5 days

Table 7. End to End Training time for Large-T model on CS-2 vs 8-GPU system

Setup	Peak Power Consumption (kW)	Total Time (h)	Total Energy Consumption (kWh)
CS-2	23.1	16.5	381.1
8-GPU	2.6	252.5	659.0

Table 8. Measured energy consumption for end-to-end training process

Conclusions

We have demonstrated the capabilities of the Cerebras CS-2 system to accelerate the training of large deep networks. We choose time to train to a specific perplexity as a comparison metric and we observe a speed up of 15x factor to achieve the same perplexity, effectively finishing training BERTLARGE in 16.47 hours.

Additionally, we have demonstrated the advantages in terms of improved accuracy and reduced perplexity in training from scratch on domain specific corpora.

In conclusion we have shown the effectiveness of the CS-2 in terms of achieving better models, being able to experiment both in terms of model size and dataset by reducing the wall clock time for experiments, while offering a reduction in energy consumed.

References

[i] Jinhyuk Lee, W. Y. (2020). BioBERT: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics.

[ii] Reuters Corpora (RCV1, RCV2, TRC2) . (n.d.). Retrieved from https://trec.nist.gov/data/reuters/reuters.html

[iii] Toutanova, J. D.-W. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

[iv] Yi Yang, Mark Christopher Siy UY, Allen Huang. "FinBERT: A Pretrained Language Model for

Financial Communications." 2020

v OpenWebText. (n.d.). Retrieved from https://github.com/jcpeterson/openwebtext

vi Alec Radford, J. W. (2018). Language Models are Unsupervised Multitask Learners.

vii Diederik P. Kingma, J. B. (2015). Adam: A Method for Stochastic Optimization. In 3rd International Conference on Learning Representations. ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings

Starting your collaboration with Cerebras

To explore how Cerebras systems can accelerate your research or to see a demo, please contact us at www.cerebras.net/get-demo.

About Cerebras

Cerebras Systems builds the fastest AI accelerators in existence.

Our systems are removing roadblocks to advances in biomedical research, energy, manufacturing, government services, and security. Our systems are doing groundbreaking work at leading institutions including GlaxoSmithKline, AstraZeneca, TotalEnergies, and Argonne National Laboratory and Lawrence Livermore National Laboratory. We offer cluster-scale acceleration in a single, easy-to-program device, so your researchers can focus on innovation, not on working around the limitation of traditional computing systems.

Learn more at www.cerebras.net.