Geek Culture

How does Batch Size impact your model learning

Different aspects that you care about

Devansh · Subscribe

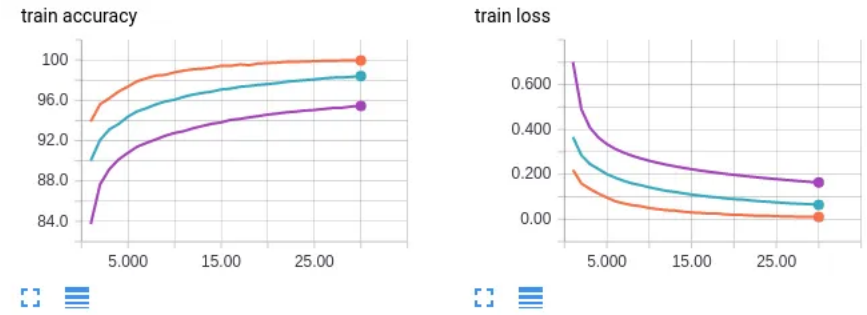
Published in Geek Culture · 7 min read · Jan 16, 2022

Join 31K+ AI People keeping in touch with the most important ideas in Machine Learning through my free newsletter over here

Batch Size is among the important hyperparameters in Machine Learning. It is the hyperparameter that defines the number of samples to work through before updating the internal model parameters. It can one of the crucial steps to making sure your models hit peak performance. It should not be surprising that there is a lot of research into how different Batch Sizes affect aspects of your ML pipelines. This article will summarize some of the relevant research when it comes to batch sizes and supervised learning. To get a complete picture of the process, we will look at how batch size affects performance, training costs, and generalization.

Training Performance/Loss

The primary metric that we care about, Batch Size has an interesting relationship with model loss. Going with the simplest approach, let's compare the performance of models where the only thing that changes is the batch size.



Training loss and accuracy when the model is trained using different batch sizes.

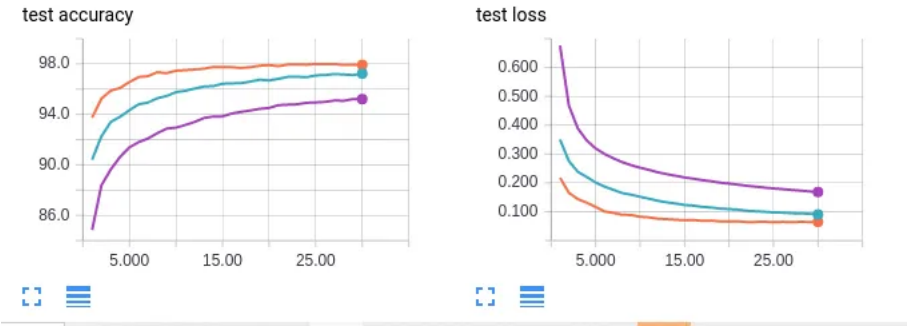
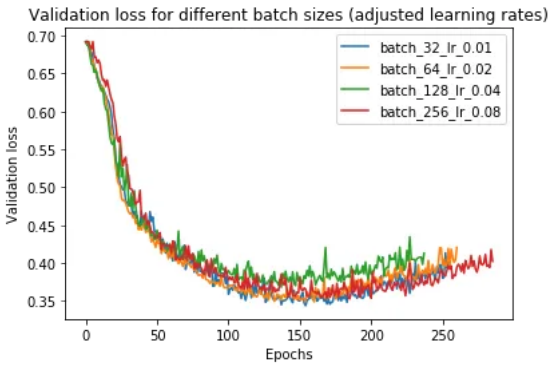


Image is taken from: https://medium.com/mini-distill/effect-of-batch-size-on-training-dynamics-21c14f7a716e#:~:text=Finding%3A%20large%20batch%20size%20means,all%20about%20the%20same%20size.

- Orange curves: batch size 64
- Blue curves: batch size 256
- Purple curves: batch size 1024

This makes it pretty clear that increasing batch size lowers performance. But it's not so straightforward. When we increase batch size, we should also adjust the learning rate to compensate for this. When we do this, we get the following result



Notice both Batch Size and lr are increasing by 2 every time

Here all the learning agents seem to have very similar results. In fact, it seems adding to the batch size reduces the validation loss. However, keep in mind that these performances are close enough where some deviation might be due to sample noise. So it's not a good idea to read too deeply into this.

The authors of, "Don't Decay the Learning Rate, Increase the Batch Size" add to this. They say that increasing batch size gives identical performance to decaying learning rate (the industry standard). Following is a quote from the paper:

> instead of decaying the learning rate, we increase the batch size during training. This strategy achieves near-identical model performance on the test set with the same number of training epochs but significantly fewer parameter updates. Our proposal does not require any fine-tuning as we follow pre-existing training schedules; when the learning rate drops by a factor of α, we instead increase the batch size by α

They show this hypothesis on several different network architectures with different learning rate schedules. This was a very comprehensive paper and I would suggest reading this paper. They came up with several steps that they used to severely cut down model training time without completely destroying performance.

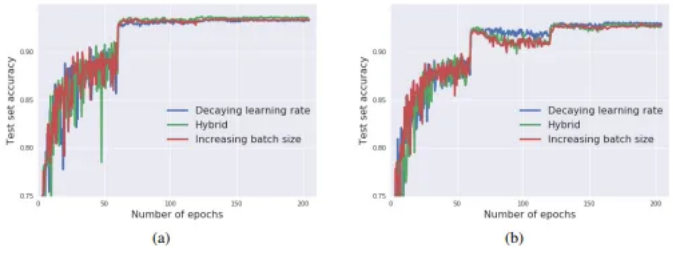


Figure 4: Wide ResNet on CIFAR10. The test set accuracy during training, for vanilla SGD (a) and Adam (b). Once again, all three schedules result in equivalent test set performance.

One of the many architectures they demonstrated their hypothesis on.

Verdict: No significant impact (as long as learning rate is adjusted accordingly).

Generalization

Generalization refers to a models ability to adapt to and perform when given new, unseen data. This is extremely important because it's highly unlikely that your training data will have every possible kind of data distribution relevant to its application.

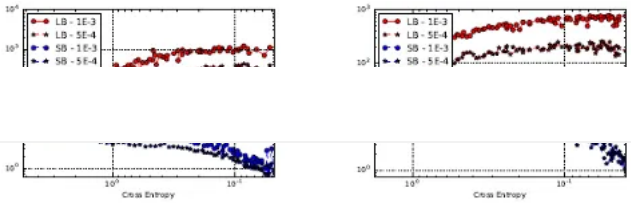


Figure 6: Sharpness v/s Cross Entropy Loss for SB and LB methods.

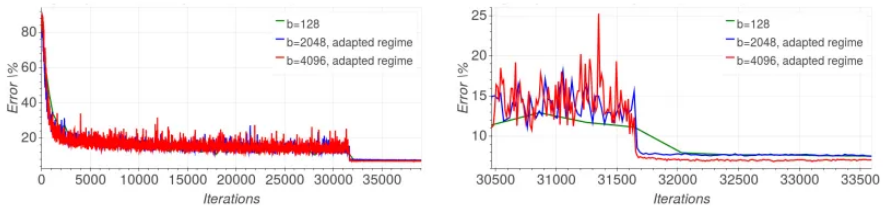
This graph shows us the sharpness of Large Batch training increases as we train (loss gets lower). The sharpness of Small Batch learners falls. This is thought to cause the generalization gap.

This is one of those areas where we see clear differences. There has been a lot of research into the difference in generalization between large and small batch training methods. The conventional wisdom states the following: *increasing batch size drops the learners' ability to generalize*. The authors of the paper, "On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima", claim that it is because Large Batch methods tend to result in models that get stuck in local minima. The idea is that smaller batches are more likely to push out local minima and find the Global Minima. If you want to read more about this paper it's takeaways read this article.

However, it doesn't end here. "Train longer, generalize better: closing the generalization gap in large batch training of neural networks" is a paper that attempts to tackle the generalization gap b/w the batch sizes. The authors make a simple claim:

> *Following this hypothesis we conducted experiments to show empirically that the "generalization gap" stems from the relatively small number of updates rather than the batch size, and can be completely eliminated by adapting the training regime used.*

Here updates refers to the number of times a model is updated. This makes sense. If a model is using double the batch size, it will by definition go through the dataset with half the updates. Their paper is quite exciting for a simple reason. If we can do away with the generalization gap, without increasing the number of updates, we can save costs while seeing a great performance.



(a) Validation error (b) Validation error - zoomed

Figure 3: Comparing generalization of large-batch regimes, adapted to match performance of small-batch training.

Here we see that once the authors used an adapted training regime, the large batch size learners caught up to the smaller batch sizes. They summarise their results in the following table:

Table 1: Validation accuracy results, SB/LB represent small and large batch respectively. GBN stands for Ghost-BN, and RA stands for regime adaptation

Network	Dataset	SB	LB	+LR	+GBN	+RA
F1 (Keskar et al., 2017)	MNIST	98.27%	97.05%	97.55%	97.60%	98.53%
C1 (Keskar et al., 2017)	Cifar10	87.80%	83.95%	86.15%	86.4%	88.20%
Resnet44 (He et al., 2016)	Cifar10	92.83%	86.10%	89.30%	90.50%	93.07%
VGG (Simonyan, 2014)	Cifar10	92.30%	84.1%	88.6%	91.50%	93.03%
C3 (Keskar et al., 2017)	Cifar100	61.25%	51.50%	57.38%	57.5%	63.20%
WResnet16-4 (Zagoruyko, 2016)	Cifar100	73.70%	68.15%	69.05%	71.20%	73.57%

Table 2: ImageNet top-1 results using Alexnet topology (Krizhevsky, 2014), notation as in Table 1.

Network	LB size	Dataset	SB	LB^8	$+LR^8$	+GBN	+RA
Alexnet	4096	ImageNet	57.10%	41.23%	53.25%	54.92%	59.5%
Alexnet	8192	ImageNet	57.10%	41.23%	53.25%	53.93%	59.5%

We see that once RA is applied, LB methods even start to surpass SB learning

This is obviously quite exciting. If we can remove/significantly reduce the generalization gap in the methods, without increasing the costs significantly, the implications are massive. If you want a breakdown of this paper, let me know in the comments/texts. I will add this paper to my list.

Verdict: Larger Batch → Weak Generalization. But this can be fixed.

Costs

This is where the Large Batch methods flip the script. Since they require a lower number of updates, they tend to pull ahead when it comes to computing power. The authors of "Don't Decay LR..." were able to reduce their training time to 30 minutes using this as one of their bases of optimization.

Machine Learning is as much engineering as it is computing

But this is not the only thing that makes a difference. And this is something that I learned recently myself. In my breakdown of the phenomenal report, "Scaling TensorFlow to 300 million predictions per second", I was surprised by a statement that the authors made. The authors said that they halved their training costs by increasing batch size. I asked about this and got the response to the left. This definitely makes sense. Especially when it comes to Big Data (like the one that the team was dealing with), such factors really blow up.

The costs side is fortunately relatively straightforward.

Verdict: Larger Batches → Fewer updates + shifting data → lower computational costs.

Closing

We see that Batch Sizes are extremely important in the model training process. This is why in most cases, you will see models trained with different batch sizes. It's very hard to know off the bat what the perfect batch size for your needs is. However, there are some trends that you can use to save time. If costs are important, LB might be your thing. SB might help when you care about Generalization and need to throw something up quickly.

Remember that we're only looking at supervised learning in this article. Things can change for other methods (like contrastive learning). Contrastive Learning seems to benefit a lot from larger batches + more epochs. To learn more about this, read this. ML is a complex field with tons to learn.

If you're preparing for interviews, this video will help you stand out

If you liked this article, check out my other content. I post regularly on Medium, YouTube, Twitter, and Substack (all linked below). I focus on Artificial Intelligence, Machine Learning, Technology, and Software Development. If you're preparing for coding interviews check out: Coding Interviews Made Simple, my free weekly newsletter. Feel free to reach out if you have any interesting projects/ideas for me as well.

For monetary support of my work following are my Venmo and Paypal. Any amount is appreciated and helps a lot. Donations unlock exclusive content such as paper analysis, special code, consultations, and reduced rates for mock interviews:

Venmo: https://account.venmo.com/u/FNU-Devansh

Paypal: paypal.me/ISeeThings

Reach out to me

If that article got you interested in reaching out to me, then this section is for you. You can reach out to me on any of the platforms, or check out any of my other content. If you'd like to discuss tutoring, text me on LinkedIn, IG, or Twitter.

Free Weekly Summary of the important updates in Machine Learning(sponsored)- https://lnkd.in/gCFTuivn

Check out my other articles on Medium. : https://rb.gy/zn1aiu

My YouTube: https://rb.gy/88iwdd

Reach out to me on LinkedIn. Let's connect: https://rb.gy/m5ok2y

My Instagram: https://rb.gy/gmvuy9

My Twitter: https://twitter.com/Machine01776819

If you're preparing for coding/technical interviews: https://codinginterviewsmadesimple.substack.com/

Get a free stock on Robinhood: https://join.robinhood.com/fnud75

Machine Learning | Artificial Intelligence | Data Science | Technology | Deep Learning

Published in Geek Culture

33K Followers · Last published Aug 31, 2023

Follow

A new tech publication by Start it up (https://medium.com/swlh).

Written by Devansh

36K Followers · 25 Following

Subscribe

Writing about AI, Math, the Tech Industry and whatever else interests me. Join my cult to gain inner peace and to support my crippling chocolate milk addiction

Responses (6)

Write a response

What are your thoughts?

Kevin Summerian
Apr 11, 2022

Another very interesting article Devansh! I kind of expected that batch size had a major impact on the training time of the model, although I'd always thought that bigger batch sizes were better at generalizing than smaller ones. My initial... more

6 1 reply Reply

Regan Yue
Jul 13, 2023

Hi, This is Regan. I am currently operating a Chinese AI blog named Baihai IDP.

Please allow me to translate this blog post into Chinese.

I am very interested in the content of your blog post. I believe that the information in it would be of great... more

3 1 reply Reply

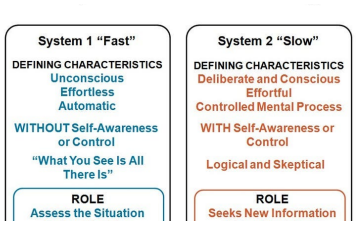
Henry John
Dec 13, 2024

Thanks for sharing, but one thing I am not really understand. Since when we do not want to sacrifice the model generation ability we need to train more steps under large batch_size [Probably even more steps then when training use small batch_size]... more

Reply

See all responses

More from Devansh and Geek Culture

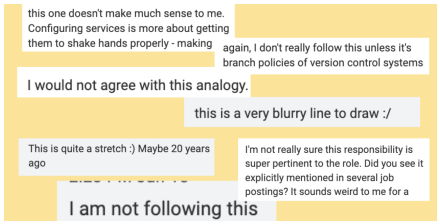


Devansh

What I Learned From Thinking Fast And Slow

Quite possibly the single most important book ever written for data scientists, decision

Apr 1, 2024 7.7K 134



In Geek Culture by Zulie Rane

I Paid a Professional to Edit a ChatGPT-Written Article. Hilarity

The results were not pretty, but very funny.

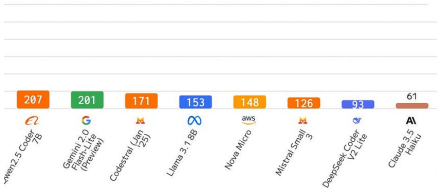
Feb 2, 2023 18.6K 417

In Geek Culture by Hasitha Subhashana

Circuit Breaker Pattern (Design Patterns for Microservices)

In a distributed system we have no idea how other components would fail. Network issues

Jun 12, 2021 906 11



Devansh

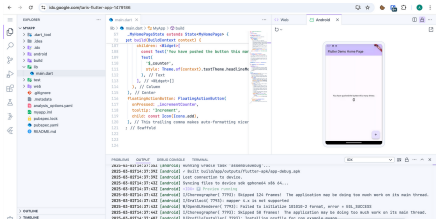
Is the Mercury LLM the first of a new Generation of LLMs?

Understanding Why Diffusion-based Generation might be the future of Language

Feb 28 204 3

See all from Devansh | See all from Geek Culture

Recommended from Medium

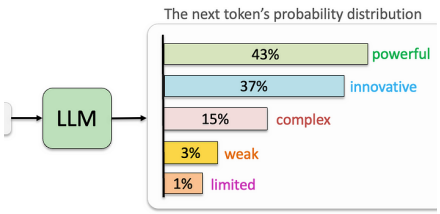


In Coding Beauty by Tari Ibaba

This new IDE from Google is an absolute game changer

This new IDE from Google is seriously revolutionary.

Mar 11 3.9K 209

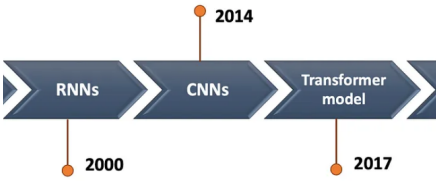


LM Po

Understanding LLM Decoding Strategies

If you're not a Medium subscriber, click here to read the full article.

Oct 25, 2024 86

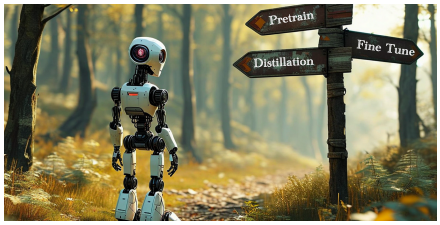


Kaouthar EL BAKOURI

ANN vs DNN

ANN and DNN for (Deep Neural Network) (Artificial Neural Network): it's a very broad

Feb 3



Justin Muller

How to choose between pre training, fine tuning, and model

A simple guide to pros and cons.

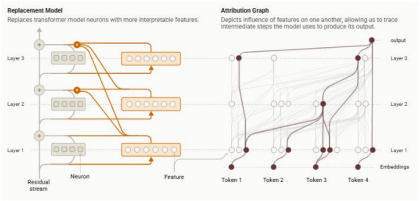
Dec 18, 2024 38

In Level Up Coding by Sahib Dhanjal

How To Train Your PyTorch Models (Much) Faster

Tips and tricks I learnt while working with the best in the industry

Feb 10 821 7



Lee Fischman

Anthropic drops an amazing report on LLM interpretability

Circuit Tracing: Revealing Computational Graphs in Language Models:

Mar 30 5

See more recommendations