HAPPYWHALE · RESEARCH PREDICTION COMPETITION · 2 YEARS AGO

Late Submission

Happywhale - Whale and Dolphin Identification

Identify whales and dolphins by unique characteristics

SANYAM BHUTANI · 614TH IN THIS COMPETITION · POSTED 2 YEARS AGO

166

9 Computer Vision Tricks to Improve Performance

Hello!

I wanted to create a post sharing some general tips & suggestions that might help improve training speeds & accuracy, I have

Without further ado:

1. Start with Smaller Resolution:

The first two tips are focused on allowing faster prototyping-the more ideas you can try, the more chances you have to score better. To iterate faster, we need to start small to keep our training times small:

Ayush has kindly created a Datasets thread here that points to all the datasets shared. Starting with a small dataset size allows you to iterate faster.

Since you'll be working with smaller GPU memory usage, you can increase `batch_size` and also iterate faster. Once you're confident in your ideas and see consistent score improvements, you can scale to larger image sizes.

2. Start with subsets of Data:

Continuing with the previous line, you should start with just a small number of classes or examples and validate your training models there.

Ex: Train on 10 classes, check if it improves CV → Submit

Scale idea to 20 classes, check CV, and submit again

oes well, train on the complete dataset.

3. Use FP16 or Half-Precision Training:

Who doesn't want up to 50% faster training?

NVIDIA GPUs have Tensor-Cores which offer huge speedups when using "Half-Precision" Tensors. I have written a more detailed blog here, the short version is to try using `fp_16` training to observe speedups on any GPU (and TPU!)

4. Use TPUs:

Kaggle offers 20 hours of TPUs every week. TPUs have 8 cores, which allow your batch_sizes to be scaled by a factor of 8. This allows for much faster training and faster iteration.

Note: I have recently discovered Hugging Face Accelerate which claims to give you easy workflow on TPUs with PyTorch too

5. Progressive Resizing:

This idea IIRC was introduced in the Efficientnet papers and also taught in the fastai courses.

Chris Deotte has a fantastic post talking about CNN Input image sizes. This blog teaches you how progressive resizing works in fastai. TL;DR:

- Train model on size: small
- Save weights and re-train model on larger image size
- Save weights again and re-train on final image sizes

process allows much faster convergence and better performance

6. Experiment: Depthwise Convs instead of Regular Convs:

I believe this concept was introduced in the MobileNet paper first and I saw it resurface in a recent discussion related to ConvNext architectures. Depthwise Convolutions have fewer filters and hence train faster.

See here for some tips on making it work in PyTorch

7. LR Scheduler:

Changing your `learning_rate` during the training of your model:

Changing your `learning_rate` during the training of your model.

A slow lr takes too long and fast lr might not help your model converge, using this logic, we should use dynamic learning rates.

There are many schedulers that allow this: I would recommend using `fastai` and its `fine_tune()` or `fit_one_cycle()` function. See here for more details.

8. LR Warmup:

This one is in-line with the previous one:

From the paper, "Bag of Tricks", one of the ticks highlights using LR warmup:

When you start training a model, it has more "randomness" as it's just starting to learn features, hence starting with a smaller `ning_rate` first allows it to pick details, and later you can increase it to the expected schedule or value after the "warmup" s are done and your model has learned some details.

View

9. Image Augmentations:

NNs benefit from more data. A slight change in an image can really help a model improve its understanding of features inside of an image.

Using correct image augmentations can really help your model. I had posted a nb sharing fastai image augmentations tutorial, I will be sharing an updated one for this competition soon if it's helpful.

Chris Deotte in his recent CTDS interview shared some secrets. Qishen Ha, whose team had won the TF GBR competition also shared some tips of making these work

TL;DR of both: Try a lot of experiments and try as many augmentations. Start with augmentations off and then add them one by one to see if your training improves.

Also, visualise results as you train models to make sure they're learning about the whales and not backgrounds!

I have two more bonus suggestions for anyone that has read this far :)

Bonus Tip #1: Use Timm or Tfimm:

Timm and Tfimm, the latter being a TF-port of the former is a fantastic resource! Ross, posts almost all the cutting edge model weights along with *extremely* optimised training methods. I would highly recommend also spending time digging into their source code but at the least using the library is a solid suggestion for anyone working on CV problems

View

s Tip #2: Use NGC Containers for Local training:

I understand many people are using Kaggle kernels and Colab for training. However, if you've invested in local hardware, Ross had taught in a thread on Twitter that the NGC Containers for PyTorch are very optimised and offer speedups

I hope you find these helpful and also find some training or score boosts! :)

Happy Kaggling!

GPU TPU CNN Computer Vision Deep Learning

28 Comments 7 appreciationcomments

Hotness

Comment here. Be patient, be friendly, and focus on ideas. We're all here to learn and improve!

This comment will be made public once posted.

View

Post Comment

Yeakub Sadlil
Posted 9 months ago

2

Your observation is excellent @init27

Reply

SubhenduRanjanMishra
Posted 2 years ago

1

One more idea about **Image Augmentations** is that , you can also use `Image Augmentations` during inference. Its very helpful for Objcet Detection tasks. I have found that by using few variants of validation image and then combining all the predicted bouding boxes, I could extract much better results from the same model.

Reply

DeepUnderstanding Posted 2 years ago · 563rd in this Competition

1

I think you are referring to TTA(Test Time Augmentation). In my experiments TTA doesn't work every time, if your data is too noisy.

View

Reply

CroDoc
Posted 2 years ago

1

Nice work! I wish there more posts that share a lot of tricks in a single thread.

Reply

Sanyam Bhutani Posted 2 years ago · 614th in this Competition TOPIC AUTHOR

1

Thank you, CroDoc! 🙏

I've found many posts scattered across different competitions, I'll try my best to collect ideas like so or maybe even livestream with some chai if time permits.

These tricks are however really simple compared to the incredible ones shared by Kagglers in competitions

Reply

View

CroDoc Posted 2 years ago

0

Ask @cdeotte to be on Chai again and share the best general tricks for computer vision 😸

Reply

DeepUnderstanding
Posted 2 years ago · 563rd in this Competition

2

Great points @init27 as always!

but I am quite skeptical of your 2nd point, that is use less classes then use more. I don't think it is a good idea, I think it will confuse the model when we will introduce more number of classes later.

Although I am not sure, can you a bit describe it more. Thanks

Reply

Sanyam Bhutani Posted 2 years ago · 614th in this Competition TOPIC AUTHOR

3

Thank you!

View

Actually, maybe I didn't I explain it well-I'm not suggesting doing transfer learning with the weights.

I was envisioning this:

Let's say I want to try model X + Y tricks/augmentations/preprocessing: start on just 10 classes and submit to LB.

Then train the same pipeline (no transfer learning, train again) on 15 classes, resubmit-is there an improvement? Yes, then that means the idea works and ideally, this would really speed up the iteration speed compared to training on a large number of classes.

This is again a small suggestion to change your iteration speed from training on complete dataset to the speed required to train on subsets.

Reply

DeepUnderstanding Posted 2 years ago · 563rd in this Competition

0

Ahh now I get it, thanks for the explanation, I think I should apply this.

Reply

Nithiyashree V K
Posted a month ago

View

0

Useful Info. Thanks for sharing these highly informative points.

Reply

HanHungHsun
Posted a year ago

0

Thanks for sharing. It is very helpful !

Reply

Ahsan Zafar Mehmood
Posted a year ago

0

Informative, I will apply in future.

Reply

View

Mehdi Elion
Posted a year ago

Really useful tips, thanks for sharing !

Reply

Dev Khant
Posted 2 years ago · 485th in this Competition

Thank you thank you so much for sharing it. Really Helpful!!!!

Reply

Nouran Elsayed(target)
Posted 2 years ago

Interesting post. Thanks for sharing..:)

Reply

View

Fozan
Posted 2 years ago

Thanks for sharing these tricks and tips , it will really help us @init27

Reply

Aadil Hussain
Posted 2 years ago

Interesting post. Thanks for sharing..:)

Reply

Junming Gao
Posted 2 years ago

Amazing tips!

Reply

View

Ishan Mehta115
Posted 2 years ago

Thanks for sharing these tricks. @init27 👍

Reply

This comment has been deleted.

This comment has been deleted.

Appreciation (7)

cwh094 Posted 2 years ago · 823rd in this Competition

Thanks for sharing

View

younis shaik Posted 2 years ago

Thanks for sharing

BoxuanZhang Posted 2 years ago

Thanks for sharing.

İlker Kara Posted 2 years ago

Thanks for sharing.

0

Mạnh Đỗ Posted 2 years ago · 471st in this Competition

Thanks for sharing! 💯

0

Moyashii Posted 2 years ago · 530th in this Competition

I learned a lot. Thank you!

0

Artem Burenok Posted 2 years ago

Thanks for these tricks.

0

View