Carnegie Mellon University

11785 - Introduction To Deep Learning

Recitation 4 - Hyperparameter Tuning Methods, Normalizations, Ensemble Methods, Study Groups

By:
Samruddhi Pai
Moayad Elamin

Hyperparameter Tuning

Learnings from Recitation 2 …..

- Required for improving accuracy, faster convergence, resource constraints, climb higher on the leaderboard 😜
- Saw various hyperparameters (eg. lr, optimizers, dropout, etc.) and what values can be tuned for those
- Other techniques like early stopping, gradient clipping, augmentations, mixed precision training…

Carnegie Mellon University

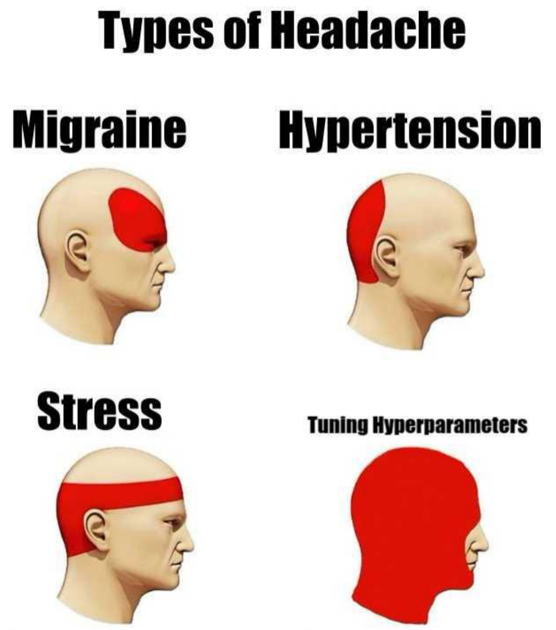
Strategies of Hyperparameter Tuning

Babysitting

- One model lots of fine tuning

Caviar

- Lots of models and parallel fine tuning



Reference - https://medium.com/@dyllanjrusher00/dont-get-stuck-tuning-hyperparameters-c9d2b29f004d

Carnegie Mellon University

Compute Requirement Considerations

- The babysitting approach requires less compute resulting in more time spent whereas, the caviar approach requires much more compute resources (which students typically don't have at their disposal)

- One suggested trade-off (given the fact most of us rely on Colab Pro or AWS g4dn's) - train baseline models in different configurations first and then shortlist the promising models, and further tune them

- If you do have unlimited compute, there are several tools that help you run grid searches or Bayesian Optimization (eg. Ray Tune)

- Practical work-around: dividing hyperparameter search among your study group to optimize compute and time

Carnegie Mellon University

Effectively Logging Experiments

Image reference - https://medium.com/swlh/hyperparameter-tuning-with-mlflow-tracking-b67ec4de18c9

Effectively Logging Experiments

Michael Crichton - 'If you don't know history, then you don't know anything. You are a leaf that doesn't know it is a part of a tree'

- It is mandatory that you log all your experiments in a very organized fashion so that you can build on your past learnings and they also help develop your intuition

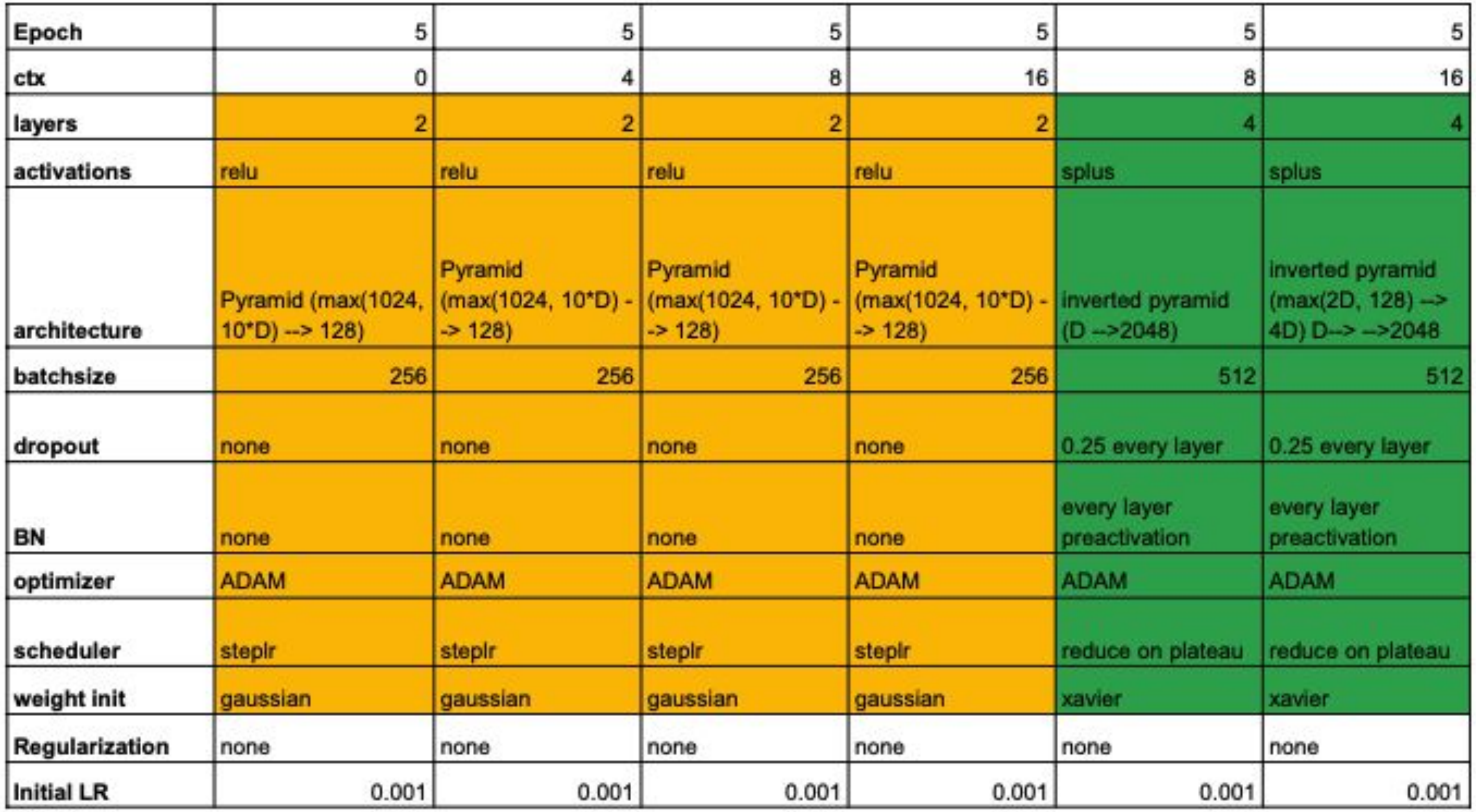
- 2 suggested strategies:

1. manually, using Google Sheets/ Excel,
2. semi-automated fashion using wandb.ai (covered in Recitation 2)

PLEASE SAVE MODELS AT EVERY EPOCH!!!!

Carnegie Mellon University

Effectively Logging Experiments - Sheets



Epoch	5	5	5	5	5	5
ctx	0	4	8	16	8	16
layers	2	2	2	2	4	4
activations	relu	relu	relu	relu	splus	splus
architecture	Pyramid (max(1024, 10*D) --> 128)	Pyramid (max(1024, 10*D) --> 128)	Pyramid (max(1024, 10*D) --> 128)	Pyramid (max(1024, 10*D) --> 128)	inverted pyramid (D -->2048)	inverted pyramid (max(2D, 128) --> 4D) D--> -->2048
batchsize	256	256	256	256	512	512
dropout	none	none	none	none	0.25 every layer	0.25 every layer
BN	none	none	none	none	every layer preactivation	every layer preactivation
optimizer	ADAM	ADAM	ADAM	ADAM	ADAM	ADAM
scheduler	steplr	steplr	steplr	steplr	reduce on plateau	reduce on plateau
weight init	gaussian	gaussian	gaussian	gaussian	xavier	xavier
Regularization	none	none	none	none	none	none
Initial LR	0.001	0.001	0.001	0.001	0.001	0.001

Effectively Logging Experiments - Sheets (Contd.)

- You should include all hyperparameters and track them in Sheets. Color code information for easy understanding and learning. Note that this is only an example. Can add other rows like inference, data-augmentation, etc.

- Start training simpler models with suggested hyperparameters then follow a methodical approach based on the results you observe

- This method has an advantage that is helps you develop intuition easily since you are tracking all hyperparameters yourself

- The obvious disadvantage is you have to manually enter the information, but the time is worth the rewards

Carnegie Mellon University

Inference from Past Expts.

- Hyperparameter tuning is one of the most important part in any Deep Learning pipeline. Depending on the task at hand and the dataset, hyperparameters need to be very carefully tuned to get good results for any problem statement

- You will get a taste of this during the Part 2's of every homework assignment and you would have seen the competitive nature of the leaderboard for HW1P2

- However, please note that hyperparameter tuning is NOT AN EXACT SCIENCE. More often than not, you will work with empirical evidence to guide your choice of hyperparameters. This makes it crucial that you stay up to date with recent research papers and you discuss with your study group

Inference from Past Expts. (Contd.)

- Deep Learning also has a human learning component which is manual hyperparameter tuning. You can operated based on domain knowledge specific intuition as well as your own experiments

- Do not form opinions based on 1 failed experiment. Even if you keep all other hyperparameters same and vary only 1, it is still not necessarily a direct effect of tuning it

- Do not vary architectures in successive experiments by a huge factor, instead progressively build on your experiments. Verify your learnings at each step, that is truly the knowledge you will gain from this course.

Carnegie Mellon University

Domain specific intuition and hyperparameters

- Domain knowledge is important and could help you build models efficiently

- It is of utmost importance to understand the task you are training the model for as well as the domain for that particular task.

- If you don't have these intuitions:
    - Read papers
    - Work with your study group

Carnegie Mellon University

Example with HW1P2

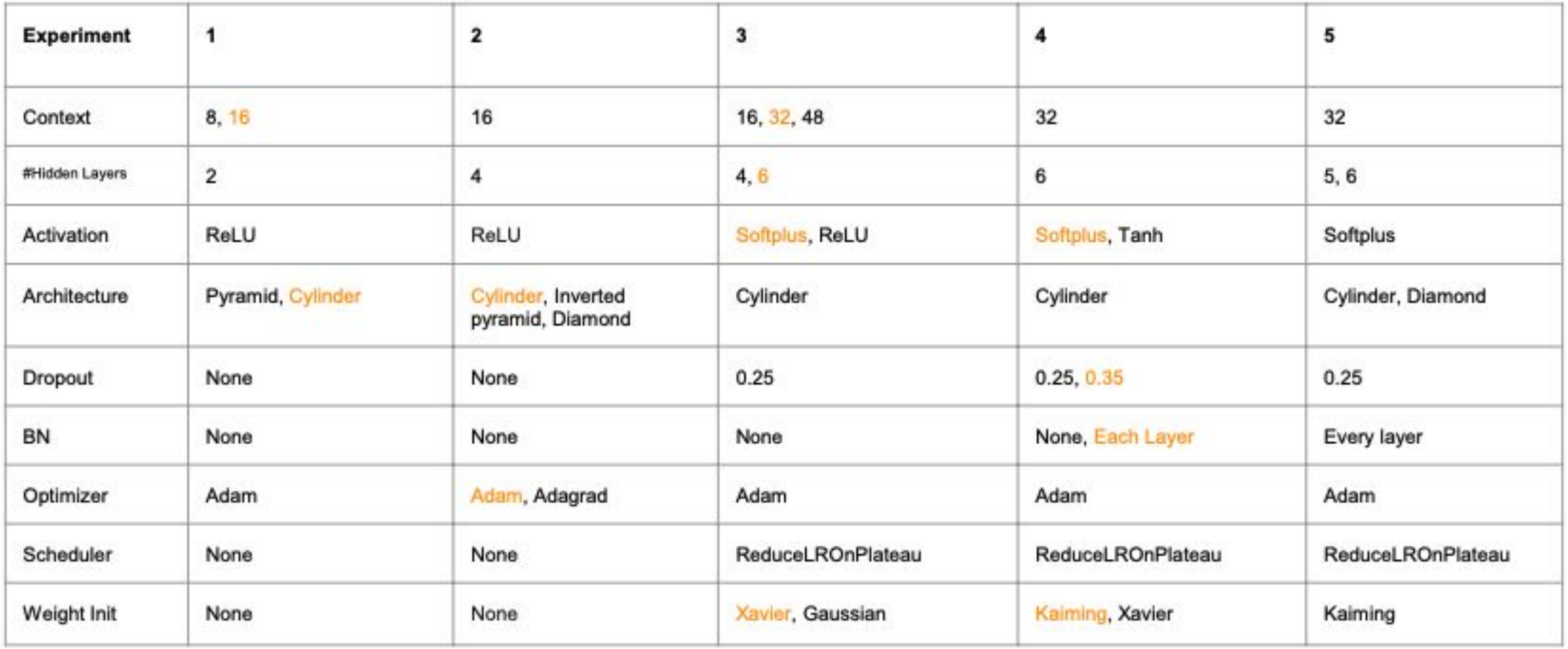
- Domain intuition: phoneme classification => context is important

- Start your work with several simple architectures, find the best parameter combination

- Change your parameter settings to see if there is any improvement

Carnegie Mellon University

Example with HW1P2



Experiment	1	2	3	4	5
Context	8, 16	16	16, 32, 48	32	32
#Hidden Layers	2	4	4, 6	6	5, 6
Activation	ReLU	ReLU	Softplus, ReLU	Softplus, Tanh	Softplus
Architecture	Pyramid, Cylinder	Cylinder, Inverted pyramid, Diamond	Cylinder	Cylinder	Cylinder, Diamond
Dropout	None	None	0.25	0.25, 0.35	0.25
BN	None	None	None	None, Each Layer	Every layer
Optimizer	Adam	Adam, Adagrad	Adam	Adam	Adam
Scheduler	None	None	ReduceLROnPlateau	ReduceLROnPlateau	ReduceLROnPlateau
Weight Init	None	None	Xavier, Gaussian	Kaiming, Xavier	Kaiming

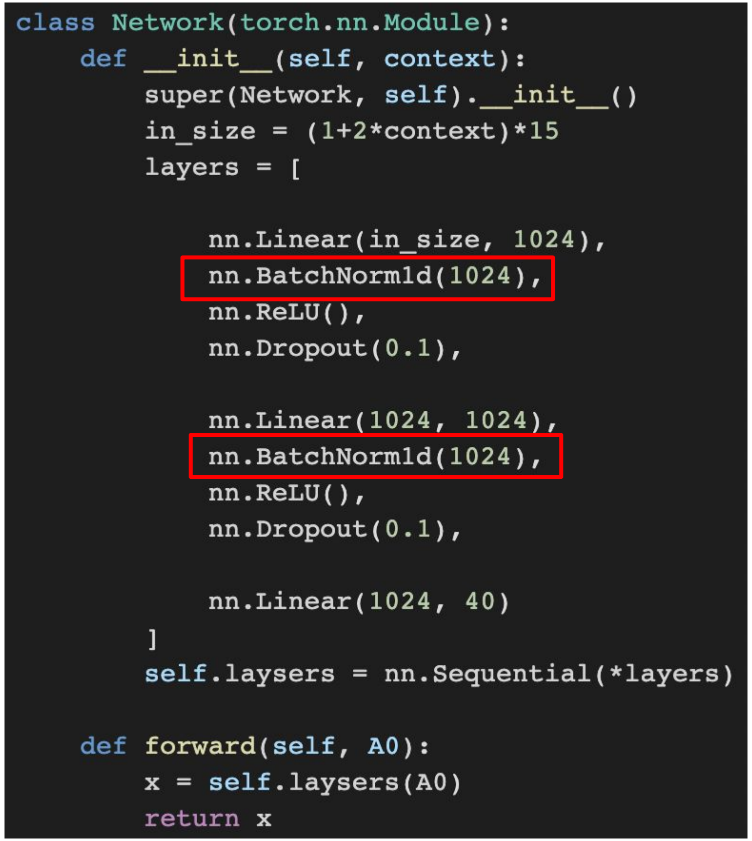
Carnegie Mellon University

Normalizations

- Batch Norm (paper),
- Layer Norm (paper),
- Weight Norm (paper),
- Instance Norm (paper),
- Group Norm (paper),
- Batch-Instance Norm (paper),
- Switchable Norm (paper)

Batch Norm

- Normalizing batch of inputs to eliminate internal covariate shift
- torch.nn.BatchNorm1d(num_features) - Applies Batch Normalization over a 2D or 3D input
- For 4D inputs (N,C,H,W) - torch.nn.BatchNorm2d(num_features)



```
class Network(torch.nn.Module):
    def __init__(self, context):
        super(Network, self).__init__()
        in_size = (1+2*context)*15
        layers = [

            nn.Linear(in_size, 1024),
            nn.BatchNorm1d(1024),
            nn.ReLU(),
            nn.Dropout(0.1),

            nn.Linear(1024, 1024),
            nn.BatchNorm1d(1024),
            nn.ReLU(),
            nn.Dropout(0.1),

            nn.Linear(1024, 40)
        ]
        self.laysers = nn.Sequential(*layers)

    def forward(self, A0):
        x = self.laysers(A0)
        return x
```

Carnegie Mellon University

Layer Norm

- Problems with Batch Norm
  - Poor performance if the batch size is small, possible for high dimensional inputs
  - Running mean and variance might not be the best thing to calculate for sequential algorithms like RNNs
- Layer normalization calculates mean and variance for each item within the batch
- Pytorch syntax - torch.nn.LayerNorm(normalized_shape)

Carnegie Mellon University

Some other methods

- Weight Norm
  - Normalizes weights of each layer torch.nn.utils.weight_norm(nn.Linear(20, 40), name='weight')
- Group Norm
  - Applies normalization over a mini-batch of inputs but splitted in groups of size num_channels/num_groups
  - torch.nn.GroupNorm(num_groups, num_channels)
- Instance Norm
  - Calculates normalization parameters across individual channels/features for each input
  - torch.nn.InstanceNorm1d(num_features)
  - torch.nn.InstanceNorm2d(num_features)

References:

1. *Normalization techniques*
2. *PyTorch nn documentation*

Carnegie Mellon University

Story of Norms

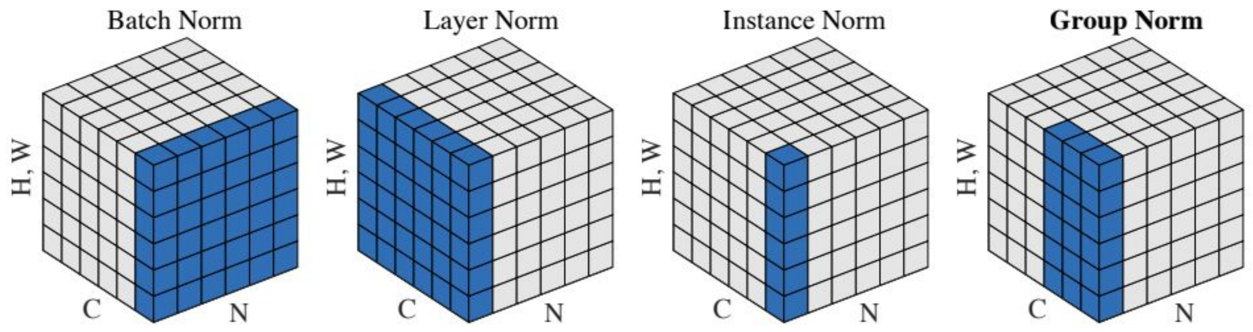


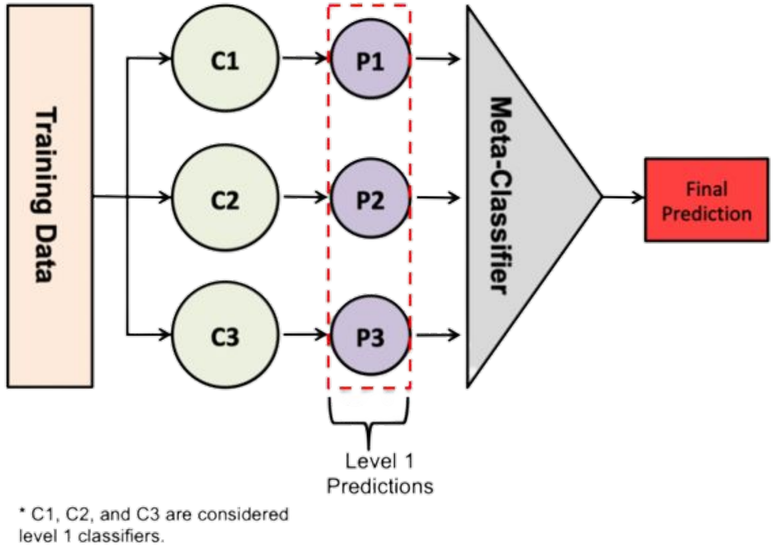
Figure 2. **Normalization methods**. Each subplot shows a feature map tensor, with N as the batch axis, C as the channel axis, and (H, W) as the spatial axes. The pixels in blue are normalized by the same mean and variance, computed by aggregating the values of these pixels.

*Image is taken from group norm paper

Carnegie Mellon University

Ensembles

- Ensemble learning is a machine learning paradigm where multiple learners are trained to solve the same problem. In contrast to ordinary machine learning approaches which try to learn one hypothesis from training data, ensemble methods try to construct a set of hypotheses and combine them to use.

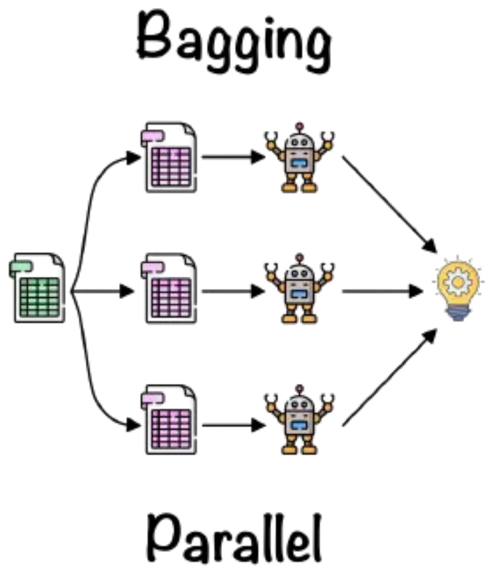


Carnegie Mellon University

Bagging

Considers homogeneous weak learners, learns them independently from each other in parallel and combines them following some kind of deterministic averaging process.

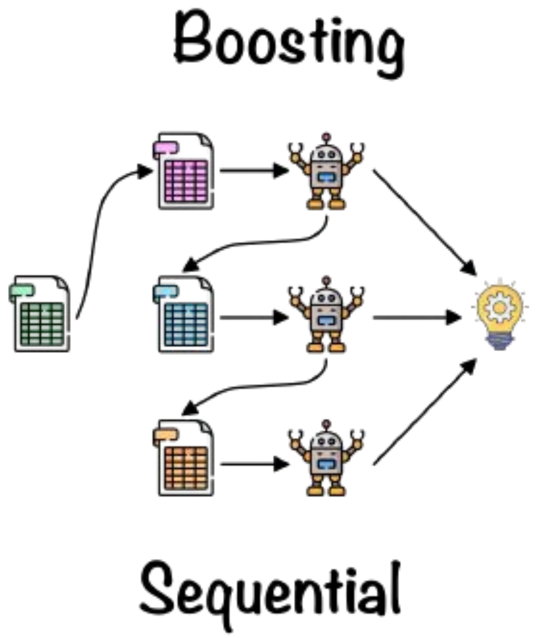
Decrease variance



Carnegie Mellon University

Boosting

Considers homogeneous weak learners, learns them sequentially in a very adaptative way (a base model depends on the previous ones) and combines them following a deterministic strategy. **Decrease bias.**



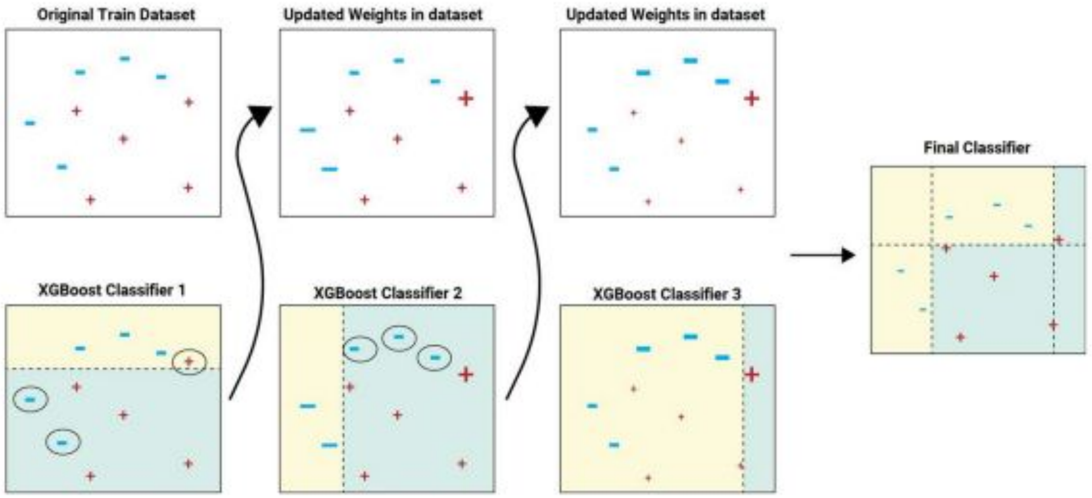
XGBoost

XGBoost as a boosting method is a category of ensemble methods. Rather than training all of the models in isolation of one another, boosting trains models in succession, with each new model being trained to correct the errors made by the previous ones. Models are added sequentially until no further improvements can be made, that’s why it is called additive model.

XGBoost: One of most powerful weapons in Kaggle. Many deep learner use it reached top in many competitions

Carnegie Mellon University

XGBoost


Original Train Dataset
Updated Weights in dataset
Updated Weights in dataset
Final Classifier
XGBoost Classifier 1
XGBoost Classifier 2
XGBoost Classifier 3

ENSEMBLES
ME TRYING TO REACH HIGH CUTOFF
20M PARAMETER LIMIT

Getting the most of Study Groups

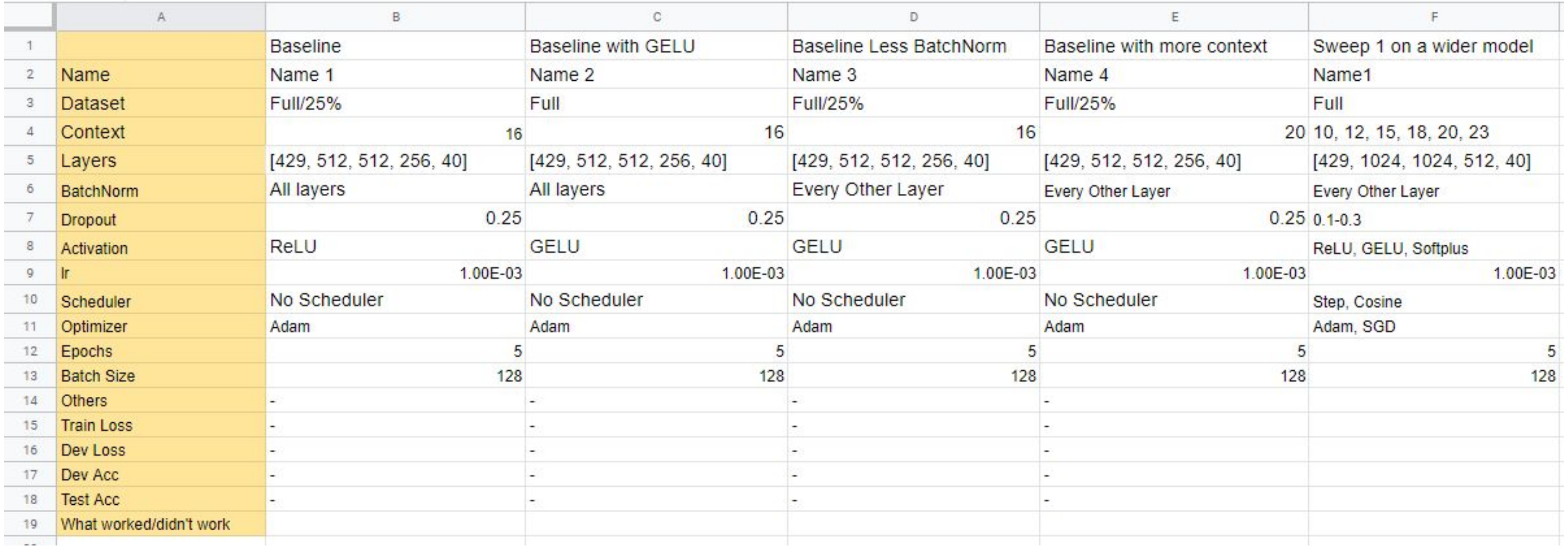
- **Why study groups?**
  - Get more from the course
  - Move towards caviar hyperparameter tuning
  - Get to High Cutoffs!!
- **How to get the most out of them?**
  - Contact each other
  - Contact your mentors
  - Join the slack channels
  - Start creating ablation sheets
  - Meet regularly!!!

Trying each and every single way to get to the high cutoff alone

Working with my study group andw all of us getting As

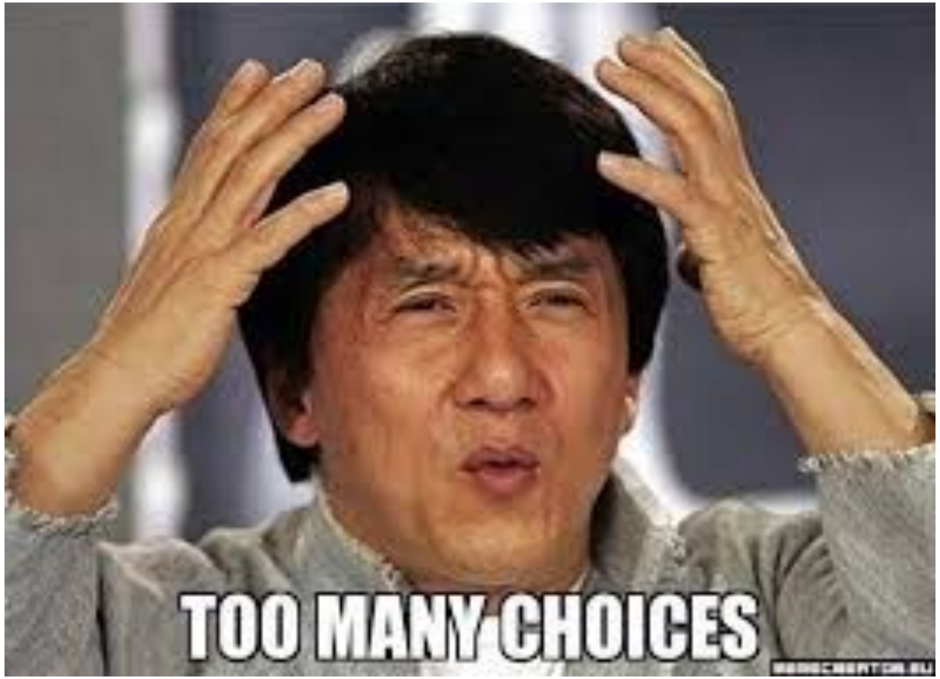
Carnegie Mellon University

A good ablation sheet



	Baseline	Baseline with GELU	Baseline Less BatchNorm	Baseline with more context	Sweep 1 on a wider model
Name	Name 1	Name 2	Name 3	Name 4	Name1
Dataset	Full/25%	Full	Full/25%	Full/25%	Full
Context	16	16	16	20	10, 12, 15, 18, 20, 23
Layers	[429, 512, 512, 256, 40]	[429, 512, 512, 256, 40]	[429, 512, 512, 256, 40]	[429, 512, 512, 256, 40]	[429, 1024, 1024, 512, 40]
BatchNorm	All layers	All layers	Every Other Layer	Every Other Layer	Every Other Layer
Dropout	0.25	0.25	0.25	0.25	0.1-0.3
Activation	ReLU	GELU	GELU	GELU	ReLU, GELU, Softplus
lr	1.00E-03	1.00E-03	1.00E-03	1.00E-03	1.00E-03
Scheduler	No Scheduler	No Scheduler	No Scheduler	No Scheduler	Step, Cosine
Optimizer	Adam	Adam	Adam	Adam	Adam, SGD
Epochs	5	5	5	5	5
Batch Size	128	128	128	128	128
Others	-	-	-	-	
Train Loss	-	-	-	-	
Dev Loss	-	-	-	-	
Dev Acc	-	-	-	-	
Test Acc	-	-	-	-	
What worked/didn't work					

Carnegie Mellon University


TOO MANY CHOICES
MEMECREATOR.EU