Supplementary Material

Kanav Anand[1]
anandkanav92@gmail.com
Ziqi Wang[1]
z.wang-8@tudelft.nl
Marco Loog[12]
M.Loog@tudelft.nl
Jan van Gemert[1]
j.c.vangemert@tudelft.nl

[1] Delft University of Technology, Delft, The Netherlands
[2] University of Copenhagen Copenhagen, Denmark

1 Hyperparameters used

The hyperparameters involved in the experiment are divided into two categories, mandatory and non-mandatory. The mandatory hyperparameters are necessary and required to train the model where as non-mandatory hyperparameters are subjected to the choice of your optimizer algorithm. If not specified, the default values for these non-mandatory hyperparameters are used to train the model. However, dividing these hyperparameters into these categories does not necessarily mean that mandatory hyperparameters are more important and impact the performance of the model more than other category. A brief information and default-values (used by PyTorch) is provided below for each Hyperparameter:

1.1 Mandatory hyperparameters

1. Epoch: epoch is defined as the number of iterations over the dataset. Minimum value is 1.

2. Batch size: batch size is the total number of training samples present in a single batch. Minimum value is 1.

3. Loss function: Every model learn by means of a loss function. It is a method of evaluating how well the model has learnt the given data. If the prediction deviates too much from truth results, loss function is usually high. There are different types of loss function available to choose from and each options is explained below:

   (a) Cross entropy: Cross entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss increases as the predicted probability diverges from the actual label. So predicting a probability of 0.1 when the actual observation label is 1 would be bad and result in a high loss value. A perfect model would have a log loss of 0.

   (b) L1 loss: It is also known as L1-norm loss function. L1 loss is basically minimizing the sum of the absolute differences between the target value and the estimated values. A perfect model would have a log loss of 0.

   (c) Mean squared loss: It is also known as L2-norm loss function. It is basically minimizing the sum of the square of the differences between the target value and the estimated values.

   (d) Negative likelihood: The negative log-likelihood becomes high at smaller values, where it can reach infinite loss, and becomes less at larger values.

4. Optimizer algorithm: During the training process, we tweak and change the parameters (weights) of our model to try and minimize that loss function, and make our predictions as correct as possible. But how exactly do you do that? How do you change the parameters of your model, by how much, and when? Optimizer together with the loss function and model parameters by updating the model in response to the output of the loss function. In simpler terms, optimizer's shape and mold your model into its most accurate possible form. The available choices for the Optimizer are explained below:

   (a) Adam optimizer

   (b) Adadelta

   (c) Averaged stochastic gradient

   (d) RMSprop

   (e) Stochastic gradient

   (f) Adagrad

1.2 Non-Mandatory hyperparameters

1. Learning rate: Learning rate is a hyper-parameter that controls how much we are adjusting the weights of our network with respect the loss gradient. It is used by all optimization algorithms mentioned above. (Default value: 0.001)

2. Weight decay: When training neural networks, it is common to use weight decay, where after each update, the weights are multiplied by a factor slightly less than 1. This prevents the weights from growing too large, and can be seen as gradient descent on a quadratic regularization term. It is used by all optimization algorithms mentioned above. (Default value: 0)

3. Rho: Rho is a coefficient used for computing a running average of squared gradients. (Default value: 0.9)

4. Lambda: Lambda is used as a decay term for gradient update. (Default value: 0)

5. Alpha: Alpha is used as a smoothing constant.(Default value: 0.99)

6. Epsilon: Sometimes the value of gradient could be really close to 0. Then, the value of the weights could blow up. To prevent the gradients from blowing up, epsilon could be included in the denominator. (Default value: 0.00001)

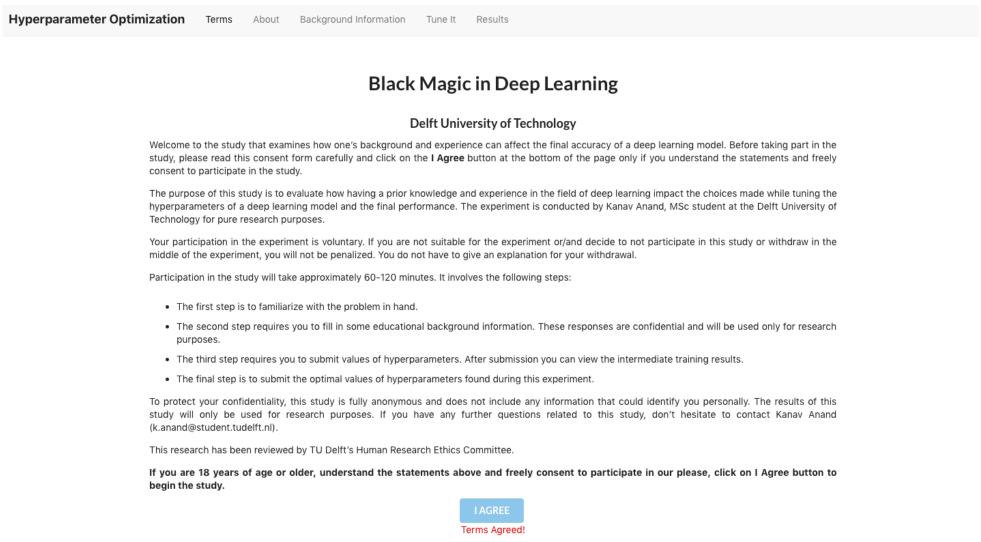
7. Momentum: Momentum helps accelerate gradients vectors in the right directions, thus leading to faster converging. (Default value: 0)

8. Learning rate decay: It specifies the rate of decay for your learning rate. (Default value: 0)

9. Initial accumulator value: It defines the starting values for accumulate gradient and accumulate updates. (Default value: 0)

10. Beta1 and Beta2: These are the coefficients used for computing running averages of gradient and its square. (Default value: 0.9, 0.999)

2 User study screenshots



Black Magic in Deep Learning

Delft University of Technology

Welcome to the study that examines how one's background and experience can affect the final accuracy of a deep learning model. Before taking part in the study, please read this consent form carefully and click on the **I Agree** button at the bottom of the page only if you understand the statements and freely consent to participate in the study.

The purpose of this study is to evaluate how having a prior knowledge and experience in the field of deep learning impact the choices made while tuning the hyperparameters of a deep learning model and the final performance. The experiment is conducted by Kanav Anand, MSc student at the Delft University of Technology for pure research purposes.

Your participation in the experiment is voluntary. If you are not suitable for the experiment or/and decide to not participate in this study or withdraw in the middle of the experiment, you will not be penalized. You do not have to give an explanation for your withdrawal.

Participation in the study will take approximately 60-120 minutes. It involves the following steps:

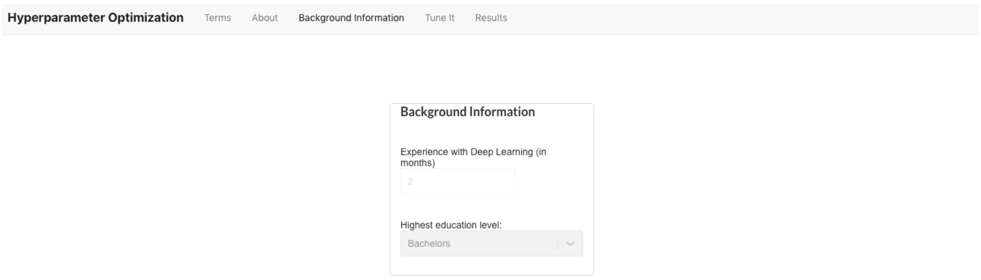
- The first step is to familiarize with the problem in hand.
- The second step requires you to fill in some educational background information. These responses are confidential and will be used only for research purposes.
- The third step requires you to submit values of hyperparameters. After submission you can view the intermediate training results.
- The final step is to submit the optimal values of hyperparameters found during this experiment.

To protect your confidentiality, this study is fully anonymous and does not include any information that could identify you personally. The results of this study will only be used for research purposes. If you have any further questions related to this study, don't hesitate to contact Kanav Anand (k.anand@student.tudelft.nl).

This research has been reviewed by TU Delft's Human Research Ethics Committee.

If you are 18 years of age or older, understand the statements above and freely consent to participate in our please, click on I Agree button to begin the study.

Figure 1: The terms and condition webpage used for the user study. It clearly states all expectations and requirements for participants.



Background Information

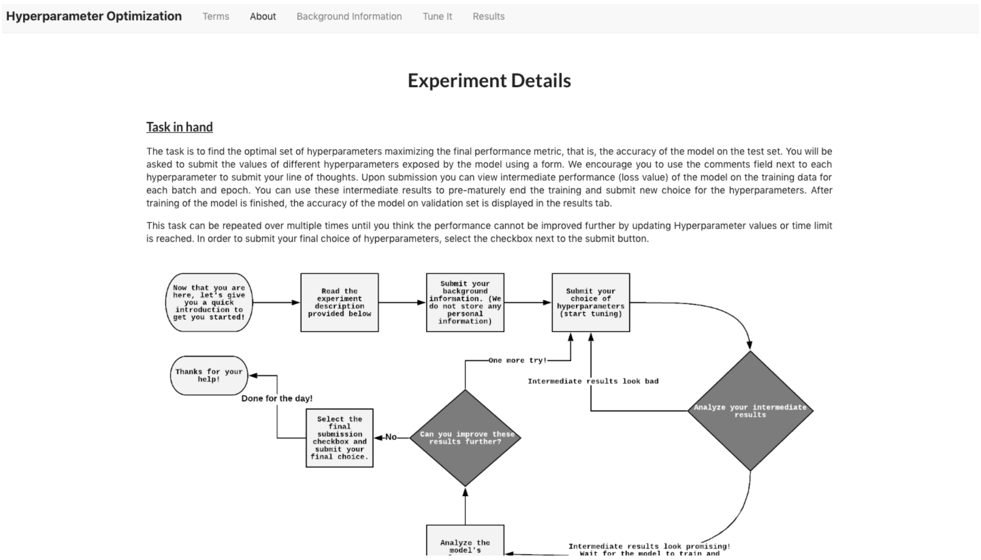
Experience with Deep Learning (in months)

2

Highest education level:

Bachelors

Figure 2: The background information page is shown above. It asks the participants to enter their relevant experience in the field of deep learning and their highest completed education level. Once the information is submitted by the user, anonymous user id is generated to identify every unique participant.


Hyperparameter Optimization
Terms
About
Background Information
Tune It
Results
Now that you are here, let's give you a quick introduction to get you started!
Read the experiment description provided below
Submit your background information. (We do not store any personal information)
Submit your choice of hyperparameters (start tuning)
Analyze your intermediate results
Intermediate results look bad
One more try!
Can you improve these results further?
No
Select the final submission checkbox and submit your final choice.
Done for the day!
Thanks for your help!
Analyze the model's
Intermediate results look promising! Wait for the model to train and

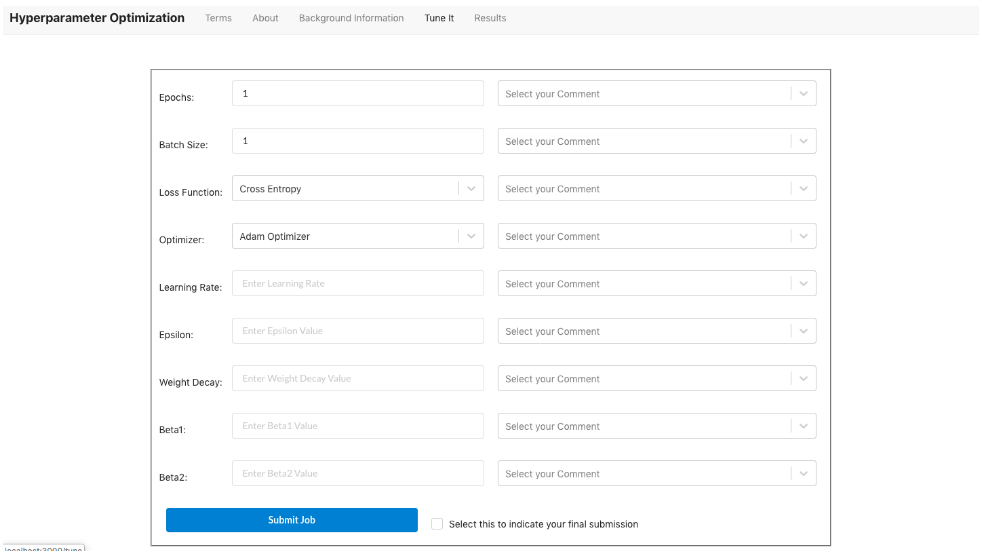
Experiment Details

Task in hand

The task is to find the optimal set of hyperparameters maximizing the final performance metric, that is, the accuracy of the model on the test set. You will be asked to submit the values of different hyperparameters exposed by the model using a form. We encourage you to use the comments field next to each hyperparameter to submit your line of thoughts. Upon submission you can view intermediate performance (loss value) of the model on the training data for each batch and epoch. You can use these intermediate results to pre-maturely end the training and submit new choice for the hyperparameters. After training of the model is finished, the accuracy of the model on validation set is displayed in the results tab.

This task can be repeated over multiple times until you think the performance cannot be improved further by updating Hyperparameter values or time limit is reached. In order to submit your final choice of hyperparameters, select the checkbox next to the submit button.

Figure 3: The experiment details page provides the detailed information of the user study. It starts with the flow diagram of the user study and follows it up with the explanation of every hyperparameter used for the experiment.



Epochs:	1	Select your Comment
Batch Size:	1	Select your Comment
Loss Function:	Cross Entropy	Select your Comment
Optimizer:	Adam Optimizer	Select your Comment
Learning Rate:	Enter Learning Rate	Select your Comment
Epsilon:	Enter Epsilon Value	Select your Comment
Weight Decay:	Enter Weight Decay Value	Select your Comment
Beta1:	Enter Beta1 Value	Select your Comment
Beta2:	Enter Beta2 Value	Select your Comment

☐ Select this to indicate your final submission

Figure 4: The hyperparameter submission form is used by the participant to submit the values for hyperparameters and their corresponding comments. Once a hyperparameter configuration is submitted, the user can view intermediate results and early stop the training of the model.