1. Pick two different emotion classes for your model to predict (e.g., anger and joy). Load/filter your dataset to include only the related class data. Create another dataset and change only one of the classes (e.g., anger and sadness) this time.

We pick “anger” and ”joy” classes in this exercise for the first dataset, and “anger”, “joy” classes for another dataset.

2.Report the combinations and corresponding results (accuracy and F1-macro) on the development set in a table:

The corresponding results of combinations are as follow (anger and joy):

Combination	Learning rate	batch_size	optimizer	Training Accuracy	F1-macro
1	0.001	128	SGD	0.9995256166982922	0.9994681354337106
2	0.01	128	SGD	0.99573055028463	0.9952165595989898
3	0.01	64	SGD	0.9971537001897534	0.9968032028631656
4	0.001	64	Adam	1.0	1.0
5	0.01	64	Adam	0.9938330170777988	0.9930953908887077
6	0.01	128	Adam	0.9985768500948766	0.99840551986633
7	0.001	64	SGD	1.0	1.0
8	0.001	128	Adam	1.0	1.0

Best Parameters: {'lr': 0.001, 'batch_size': 64, 'optimizer': <class 'torch.optim.adam.Adam'>}
Best Accuracy: 1.0, Best F1 Macro: 1.0

3. Use your best-performing model settings to train another model on the second dataset. Report the model performance (accuracy and F1-macro) on the test set of both datasets.

anger and joy:

Combination	Learning rate	batch_size	optimizer	Test Accuracy	F1-macro
1	0.001	64	Adam	0.5807860262008734	0.5249655619479783

anger and sadness:

Combination	Learning rate	batch_size	optimizer	Test Accuracy	F1-macro
1	0.001	64	Adam	0.4968085106382979	0.4774334509819132

4. What could be the reason that the specific combination/values of hyperparameters resulted in the best model performance? Don’t worry about the exact reasoning, the goal is just to provide educated (or well- reasoned) guesses.

Learning rate: The learning rate is a critical hyperparameter that influences the convergence and optimization trajectory of a model. In the context of the provided results, the optimal learning rate seems to be 0.01. A larger learning rate facilitates quicker convergence as the model takes larger steps in parameter space. However, this larger step size can lead to overshooting, causing the optimization process to oscillate or diverge. Conversely, a smaller learning rate allows the model to make finer adjustments to the parameters but requires more time to converge, particularly when dealing with extensive datasets. The choice of 0.1 strikes a balance between speed and precision, resulting in effective parameter tuning without sacrificing stability, even with a large dataset.

Batch_size: In the results, it seems that the larger batch_size(64) has a better impact on performance. The possible reasons could be that a larger batch size provides a more accurate estimate of the gradient over the entire dataset, and the smaller batches introduce more noise during training.

Optimizer: Commonly used optimizers like Adam and SGD offer different trade-offs in terms of convergence speed and stability. Adam is an adaptive optimizer that adjusts the learning rates for each parameter individually, often leading to faster convergence. On the other hand, SGD is a classic optimizer that updates parameters using a fixed learning rate. The choice between these optimizers depends on the specific characteristics of the dataset and the problem at hand. Experimenting with different optimizers and their hyperparameters can help identify the most suitable optimization strategy for a given task.

###Error encountered:
1) Issue with PyTorch and Grid Search: After constructing the model using PyTorch, we encountered a limitation with implementing grid search due to a TypeError stating, “estimator should be an estimator implementing 'fit' method”. It appears that a wrapper is needed to resolve this issue. Finally, we chose to use random search instead of grid search.

2) CUDA issue : When we applied our best-performing model settings to train on a second dataset, we encountered a CUDA issue. Upon thorough debugging, including checks on the data loader and tensor types, we discovered that the issue resulted in the labels being ‘0’ and ‘3’. In this task, labels should be contiguous integers starting from ‘0’ for proper functioning.