PS D:\NMIMS\Sem 5\IVP\Emotion-Detection> python main.py

2024-09-07 19:10:22.025214: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results

due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF\_ENABLE\_ONEDNN\_OPTS=0`.

2024-09-07 19:10:23.816403: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results

due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF\_ENABLE\_ONEDNN\_OPTS=0`.

Opening dataset in read-only mode as you don't have write permissions.

This dataset can be visualized in Jupyter Notebook by ds.visualize() or at https://app.activeloop.ai/activeloop/fer2013-train

hub://activeloop/fer2013-train loaded successfully.

Opening dataset in read-only mode as you don't have write permissions.

This dataset can be visualized in Jupyter Notebook by ds.visualize() or at https://app.activeloop.ai/activeloop/fer2013-public-test

hub://activeloop/fer2013-public-test loaded successfully.

Opening dataset in read-only mode as you don't have write permissions.

This dataset can be visualized in Jupyter Notebook by ds.visualize() or at https://app.activeloop.ai/activeloop/fer2013-private-test

hub://activeloop/fer2013-private-test loaded successfully.

C:\Users\Admin\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\layers\convolutional\base\_conv.py:107: UserWarning: Do not pass an `input\_shape`/`input\_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the mPS D:\NMIMS\Sem 5\IVP\Emotion-Detection> python main.py

2024-09-07 19:13:58.855745: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF\_ENABLE\_ONEDNN\_OPTS=0`.

2024-09-07 19:14:00.281905: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF\_ENABLE\_ONEDNN\_OPTS=0`.

Opening dataset in read-only mode as you don't have write permissions.

This dataset can be visualized in Jupyter Notebook by ds.visualize() or at https://app.activeloop.ai/activeloop/fer2013-train

hub://activeloop/fer2013-train loaded successfully.

Opening dataset in read-only mode as you don't have write permissions.

This dataset can be visualized in Jupyter Notebook by ds.visualize() or at https://app.activeloop.ai/activeloop/fer2013-public-test

hub://activeloop/fer2013-public-test loaded successfully.

Opening dataset in read-only mode as you don't have write permissions.

This dataset can be visualized in Jupyter Notebook by ds.visualize() or at https://app.activeloop.ai/activeloop/fer2013-private-test

hub://activeloop/fer2013-private-test loaded successfully.

C:\Users\Admin\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\layers\convolutional\base\_conv.py:107: UserWarning: Do not pass an `input\_shape`/`input\_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.

super().\_\_init\_\_(activity\_regularizer=activity\_regularizer, \*\*kwargs)

2024-09-07 19:15:43.483502: I tensorflow/core/platform/cpu\_feature\_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations.

To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.

Epoch 1/100

C:\Users\Admin\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\trainers\data\_adapters\py\_dataset\_adapter.py:122: UserWarning: Your `PyDataset` class should call `s

Your `PyDataset` class should call `super().\_\_init\_\_(\*\*kwargs)` in its constructor. `\*\*kwargs` can include `workers`, `use\_multiprocessing`, `max\_nts to `fit()`, as they will be ignorqueue\_size`. Do not pass these arguments to `fit()`, as they will be ignored.

self.\_warn\_if\_super\_not\_called()

718/718 ━━━━━━━━━━━━━━━━━━━━ 346s 472ms/step - accuracy: 0.2202 - loss: 10.8100 - val\_accuracy: 0.2560 - val\_loss: 2.0327

Epoch 2/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 397s 553ms/step - accuracy: 0.2348 - loss: 2.0309 - val\_accuracy: 0.2355 - val\_loss: 1.8855

Epoch 3/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 600s 835ms/step - accuracy: 0.2366 - loss: 1.9129 - val\_accuracy: 0.2560 - val\_loss: 1.9123

Epoch 4/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 466s 648ms/step - accuracy: 0.2515 - loss: 1.8678 - val\_accuracy: 0.3079 - val\_loss: 1.7928

Epoch 5/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 509s 707ms/step - accuracy: 0.2756 - loss: 1.8142 - val\_accuracy: 0.3067 - val\_loss: 1.7464

Epoch 6/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 426s 592ms/step - accuracy: 0.2833 - loss: 1.7978 - val\_accuracy: 0.2247 - val\_loss: 1.8618

Epoch 7/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 465s 648ms/step - accuracy: 0.3113 - loss: 1.7633 - val\_accuracy: 0.1844 - val\_loss: 1.9954

Epoch 8/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 460s 641ms/step - accuracy: 0.3439 - loss: 1.6963 - val\_accuracy: 0.3887 - val\_loss: 1.6543

Epoch 9/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 456s 635ms/step - accuracy: 0.3688 - loss: 1.6634 - val\_accuracy: 0.3715 - val\_loss: 1.7784

Epoch 10/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 438s 610ms/step - accuracy: 0.3785 - loss: 1.6440 - val\_accuracy: 0.4270 - val\_loss: 1.5662

Epoch 11/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 340s 473ms/step - accuracy: 0.3954 - loss: 1.6219 - val\_accuracy: 0.3488 - val\_loss: 1.9425

Epoch 12/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 335s 466ms/step - accuracy: 0.4048 - loss: 1.5983 - val\_accuracy: 0.3683 - val\_loss: 1.6473

Epoch 13/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 331s 460ms/step - accuracy: 0.4086 - loss: 1.5856 - val\_accuracy: 0.4013 - val\_loss: 1.7065

Epoch 14/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 336s 468ms/step - accuracy: 0.4139 - loss: 1.5769 - val\_accuracy: 0.4082 - val\_loss: 1.7167

Epoch 15/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 347s 482ms/step - accuracy: 0.4242 - loss: 1.5616 - val\_accuracy: 0.4359 - val\_loss: 1.5753

Epoch 16/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 358s 499ms/step - accuracy: 0.4241 - loss: 1.5552 - val\_accuracy: 0.2938 - val\_loss: 1.9833

Epoch 17/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 356s 495ms/step - accuracy: 0.4282 - loss: 1.5506 - val\_accuracy: 0.4105 - val\_loss: 1.5752

Epoch 18/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 346s 481ms/step - accuracy: 0.4351 - loss: 1.5445 - val\_accuracy: 0.4141 - val\_loss: 1.5879

Epoch 19/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 344s 478ms/step - accuracy: 0.4346 - loss: 1.5413 - val\_accuracy: 0.4507 - val\_loss: 1.5134

Epoch 20/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 349s 486ms/step - accuracy: 0.4430 - loss: 1.5221 - val\_accuracy: 0.3067 - val\_loss: 2.2668

Epoch 21/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 361s 503ms/step - accuracy: 0.4446 - loss: 1.5196 - val\_accuracy: 0.4274 - val\_loss: 1.5663

Epoch 22/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 330s 460ms/step - accuracy: 0.4431 - loss: 1.5121 - val\_accuracy: 0.3978 - val\_loss: 1.6242

Epoch 23/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 326s 454ms/step - accuracy: 0.4524 - loss: 1.5020 - val\_accuracy: 0.3805 - val\_loss: 1.6755

Epoch 24/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 326s 453ms/step - accuracy: 0.4576 - loss: 1.4980 - val\_accuracy: 0.4448 - val\_loss: 1.5063

Epoch 25/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 325s 453ms/step - accuracy: 0.4514 - loss: 1.4994 - val\_accuracy: 0.3873 - val\_loss: 1.6140

Epoch 26/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 331s 461ms/step - accuracy: 0.4615 - loss: 1.4835 - val\_accuracy: 0.4643 - val\_loss: 1.4774

Epoch 27/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 329s 458ms/step - accuracy: 0.4631 - loss: 1.4850 - val\_accuracy: 0.3131 - val\_loss: 1.7439

Epoch 28/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 326s 454ms/step - accuracy: 0.4611 - loss: 1.4905 - val\_accuracy: 0.4855 - val\_loss: 1.4551

Epoch 29/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 326s 454ms/step - accuracy: 0.4776 - loss: 1.4630 - val\_accuracy: 0.4937 - val\_loss: 1.4099

Epoch 30/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 329s 458ms/step - accuracy: 0.4629 - loss: 1.4753 - val\_accuracy: 0.5068 - val\_loss: 1.3740

Epoch 31/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 338s 471ms/step - accuracy: 0.4686 - loss: 1.4600 - val\_accuracy: 0.4488 - val\_loss: 1.4984

Epoch 32/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 328s 456ms/step - accuracy: 0.4750 - loss: 1.4518 - val\_accuracy: 0.4723 - val\_loss: 1.4489

Epoch 33/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 324s 452ms/step - accuracy: 0.4704 - loss: 1.4648 - val\_accuracy: 0.4674 - val\_loss: 1.4422

Epoch 34/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 335s 467ms/step - accuracy: 0.4764 - loss: 1.4614 - val\_accuracy: 0.4998 - val\_loss: 1.4019

Epoch 35/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 373s 520ms/step - accuracy: 0.4752 - loss: 1.4462 - val\_accuracy: 0.4760 - val\_loss: 1.4681

Epoch 36/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 424s 590ms/step - accuracy: 0.4704 - loss: 1.4626 - val\_accuracy: 0.4434 - val\_loss: 1.5149

Epoch 37/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 458s 638ms/step - accuracy: 0.4696 - loss: 1.4642 - val\_accuracy: 0.4488 - val\_loss: 1.5529

Epoch 38/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 482s 671ms/step - accuracy: 0.4796 - loss: 1.4448 - val\_accuracy: 0.4847 - val\_loss: 1.4239

Epoch 39/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 548s 763ms/step - accuracy: 0.4718 - loss: 1.4506 - val\_accuracy: 0.4805 - val\_loss: 1.4626

Epoch 40/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 565s 786ms/step - accuracy: 0.4824 - loss: 1.4463 - val\_accuracy: 0.5073 - val\_loss: 1.3695

Epoch 41/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 534s 742ms/step - accuracy: 0.4759 - loss: 1.4417 - val\_accuracy: 0.4685 - val\_loss: 1.4362

Epoch 42/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 501s 697ms/step - accuracy: 0.4833 - loss: 1.4396 - val\_accuracy: 0.5293 - val\_loss: 1.3417

Epoch 43/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 546s 761ms/step - accuracy: 0.4819 - loss: 1.4446 - val\_accuracy: 0.4815 - val\_loss: 1.4438

Epoch 44/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 350s 487ms/step - accuracy: 0.4778 - loss: 1.4512 - val\_accuracy: 0.3389 - val\_loss: 1.8208

Epoch 45/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 348s 485ms/step - accuracy: 0.4750 - loss: 1.4441 - val\_accuracy: 0.4349 - val\_loss: 1.5834

Epoch 46/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 331s 461ms/step - accuracy: 0.4842 - loss: 1.4357 - val\_accuracy: 0.5117 - val\_loss: 1.3621

Epoch 47/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 325s 453ms/step - accuracy: 0.4781 - loss: 1.4463 - val\_accuracy: 0.4831 - val\_loss: 1.4493

Epoch 48/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 334s 464ms/step - accuracy: 0.4879 - loss: 1.4385 - val\_accuracy: 0.4855 - val\_loss: 1.4260

Epoch 49/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 328s 456ms/step - accuracy: 0.4870 - loss: 1.4304 - val\_accuracy: 0.4934 - val\_loss: 1.4300

Epoch 50/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 327s 455ms/step - accuracy: 0.4859 - loss: 1.4296 - val\_accuracy: 0.4937 - val\_loss: 1.4181

Epoch 51/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 328s 457ms/step - accuracy: 0.4834 - loss: 1.4384 - val\_accuracy: 0.4545 - val\_loss: 1.4934

Epoch 52/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 325s 452ms/step - accuracy: 0.4830 - loss: 1.4390 - val\_accuracy: 0.4850 - val\_loss: 1.4195

Epoch 53/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 324s 451ms/step - accuracy: 0.4836 - loss: 1.4439 - val\_accuracy: 0.4974 - val\_loss: 1.4006

Epoch 54/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 327s 456ms/step - accuracy: 0.4912 - loss: 1.4350 - val\_accuracy: 0.4767 - val\_loss: 1.4585

Epoch 55/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 330s 460ms/step - accuracy: 0.4853 - loss: 1.4323 - val\_accuracy: 0.5007 - val\_loss: 1.4001

Epoch 56/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 323s 450ms/step - accuracy: 0.4865 - loss: 1.4256 - val\_accuracy: 0.4951 - val\_loss: 1.4118

Epoch 57/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 328s 457ms/step - accuracy: 0.4944 - loss: 1.4108 - val\_accuracy: 0.5117 - val\_loss: 1.3523

Epoch 58/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 328s 457ms/step - accuracy: 0.4927 - loss: 1.4275 - val\_accuracy: 0.5044 - val\_loss: 1.3943

Epoch 59/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 334s 465ms/step - accuracy: 0.4892 - loss: 1.4254 - val\_accuracy: 0.4997 - val\_loss: 1.4135

Epoch 60/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 327s 455ms/step - accuracy: 0.4833 - loss: 1.4273 - val\_accuracy: 0.4991 - val\_loss: 1.4312

Epoch 61/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 324s 451ms/step - accuracy: 0.4892 - loss: 1.4289 - val\_accuracy: 0.5233 - val\_loss: 1.3277

Epoch 62/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 331s 461ms/step - accuracy: 0.4906 - loss: 1.4223 - val\_accuracy: 0.5094 - val\_loss: 1.3597

Epoch 63/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 329s 458ms/step - accuracy: 0.4912 - loss: 1.4166 - val\_accuracy: 0.4869 - val\_loss: 1.4319

Epoch 64/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 326s 454ms/step - accuracy: 0.4921 - loss: 1.4188 - val\_accuracy: 0.4612 - val\_loss: 1.5028

Epoch 65/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 328s 457ms/step - accuracy: 0.4958 - loss: 1.4185 - val\_accuracy: 0.5221 - val\_loss: 1.3403

Epoch 66/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 325s 453ms/step - accuracy: 0.4912 - loss: 1.4215 - val\_accuracy: 0.5061 - val\_loss: 1.3831

Epoch 67/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 324s 451ms/step - accuracy: 0.4955 - loss: 1.4121 - val\_accuracy: 0.5026 - val\_loss: 1.3750

Epoch 68/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 325s 452ms/step - accuracy: 0.4917 - loss: 1.4249 - val\_accuracy: 0.5286 - val\_loss: 1.3290

Epoch 69/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 325s 453ms/step - accuracy: 0.4928 - loss: 1.4133 - val\_accuracy: 0.5244 - val\_loss: 1.3310

Epoch 70/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 332s 462ms/step - accuracy: 0.4878 - loss: 1.4229 - val\_accuracy: 0.5122 - val\_loss: 1.3448

Epoch 71/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 326s 454ms/step - accuracy: 0.4943 - loss: 1.4207 - val\_accuracy: 0.4908 - val\_loss: 1.4143

Epoch 72/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 325s 452ms/step - accuracy: 0.4883 - loss: 1.4148 - val\_accuracy: 0.4967 - val\_loss: 1.4252

Epoch 73/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 326s 453ms/step - accuracy: 0.4969 - loss: 1.4027 - val\_accuracy: 0.4568 - val\_loss: 1.4708

Epoch 74/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 325s 452ms/step - accuracy: 0.5002 - loss: 1.4145 - val\_accuracy: 0.5179 - val\_loss: 1.3460

Epoch 75/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 326s 454ms/step - accuracy: 0.4980 - loss: 1.4049 - val\_accuracy: 0.5132 - val\_loss: 1.3571

Epoch 76/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 330s 459ms/step - accuracy: 0.4936 - loss: 1.4161 - val\_accuracy: 0.4962 - val\_loss: 1.4016

Epoch 77/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 328s 457ms/step - accuracy: 0.4900 - loss: 1.4246 - val\_accuracy: 0.4892 - val\_loss: 1.4115

Epoch 78/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 326s 453ms/step - accuracy: 0.4969 - loss: 1.4093 - val\_accuracy: 0.4603 - val\_loss: 1.5983

Epoch 79/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 326s 453ms/step - accuracy: 0.4979 - loss: 1.4095 - val\_accuracy: 0.4634 - val\_loss: 1.4328

Epoch 80/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 326s 454ms/step - accuracy: 0.4950 - loss: 1.4089 - val\_accuracy: 0.5418 - val\_loss: 1.2938

Epoch 81/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 333s 464ms/step - accuracy: 0.4963 - loss: 1.4056 - val\_accuracy: 0.4558 - val\_loss: 1.6456

Epoch 82/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 329s 457ms/step - accuracy: 0.5002 - loss: 1.4033 - val\_accuracy: 0.4894 - val\_loss: 1.4690

Epoch 83/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 326s 453ms/step - accuracy: 0.5019 - loss: 1.4066 - val\_accuracy: 0.5101 - val\_loss: 1.3814

Epoch 84/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 327s 455ms/step - accuracy: 0.4949 - loss: 1.4078 - val\_accuracy: 0.5282 - val\_loss: 1.3284

Epoch 85/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 326s 453ms/step - accuracy: 0.4911 - loss: 1.4195 - val\_accuracy: 0.4922 - val\_loss: 1.4315

Epoch 86/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 327s 456ms/step - accuracy: 0.4949 - loss: 1.4172 - val\_accuracy: 0.5263 - val\_loss: 1.3291

Epoch 87/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 367s 511ms/step - accuracy: 0.4989 - loss: 1.4041 - val\_accuracy: 0.4822 - val\_loss: 1.4241

Epoch 88/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 392s 546ms/step - accuracy: 0.4897 - loss: 1.4102 - val\_accuracy: 0.5195 - val\_loss: 1.3268

Epoch 89/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 411s 571ms/step - accuracy: 0.4999 - loss: 1.4022 - val\_accuracy: 0.5150 - val\_loss: 1.3415

Epoch 90/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 408s 567ms/step - accuracy: 0.5009 - loss: 1.3996 - val\_accuracy: 0.5206 - val\_loss: 1.3524

Epoch 91/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 334s 465ms/step - accuracy: 0.5002 - loss: 1.4040 - val\_accuracy: 0.5150 - val\_loss: 1.3458

Epoch 92/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 341s 475ms/step - accuracy: 0.4991 - loss: 1.4061 - val\_accuracy: 0.5077 - val\_loss: 1.4026

Epoch 93/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 341s 474ms/step - accuracy: 0.4950 - loss: 1.4029 - val\_accuracy: 0.5138 - val\_loss: 1.3317

Epoch 94/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 340s 473ms/step - accuracy: 0.4915 - loss: 1.4167 - val\_accuracy: 0.5132 - val\_loss: 1.3713

Epoch 95/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 338s 470ms/step - accuracy: 0.5066 - loss: 1.3947 - val\_accuracy: 0.5152 - val\_loss: 1.3659

Epoch 96/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 343s 477ms/step - accuracy: 0.4979 - loss: 1.4112 - val\_accuracy: 0.5068 - val\_loss: 1.3787

Epoch 97/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 340s 474ms/step - accuracy: 0.5028 - loss: 1.3985 - val\_accuracy: 0.5136 - val\_loss: 1.3522

Epoch 98/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 327s 456ms/step - accuracy: 0.5012 - loss: 1.4017 - val\_accuracy: 0.5030 - val\_loss: 1.4113

Epoch 99/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 327s 455ms/step - accuracy: 0.4960 - loss: 1.4023 - val\_accuracy: 0.4934 - val\_loss: 1.4730

Epoch 100/100

718/718 ━━━━━━━━━━━━━━━━━━━━ 328s 456ms/step - accuracy: 0.4992 - loss: 1.4090 - val\_accuracy: 0.5110 - val\_loss: 1.3798

Saving the model...

WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save\_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my\_model.keras')` or `keras.saving.save\_model(model, 'my\_model.keras')`.

Model saved

113/113 ━━━━━━━━━━━━━━━━━━━━ 13s 113ms/step - accuracy: 0.4919 - loss: 1.3947

Test accuracy: 0.4979