

# Package ‘fastai’

October 13, 2022

**Type** Package

**Title** Interface to 'fastai'

**Version** 2.2.0

**Maintainer** Turgut Abdullayev <turqut.a.314@gmail.com>

**Description** The 'fastai' <<https://docs.fast.ai/index.html>> library simplifies training fast and accurate neural networks using modern best practices. It is based on research in to deep learning best practices undertaken at 'fast.ai', including 'out of the box' support for vision, text, tabular, audio, time series, and collaborative filtering models.

**License** Apache License 2.0

**URL** <https://github.com/EagerAI/fastai>

**BugReports** <https://github.com/EagerAI/fastai/issues>

**Encoding** UTF-8

**RoxygenNote** 7.1.2

**Imports** reticulate, generics, png, ggplot2, ggpubr, glue

**Suggests** knitr, testthat, rmarkdown, curl, magrittr, data.table, vctrs, stats, utils, R.utils, viridis, zeallot

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Turgut Abdullayev [ctb, cre, cph, aut]

**Repository** CRAN

**Date/Publication** 2022-03-21 07:50:18 UTC

## R topics documented:

*.fastai.torch_core.TensorMask . . . . .	22
+.fastai.torch_core.TensorMask . . . . .	22
+.torch.nn.modules.container.Sequential . . . . .	23

<code>/.fastai.torch_core.TensorMask</code>	23
<code>&lt;.fastai.torch_core.TensorMask</code>	24
<code>&lt;=.fastai.torch_core.TensorMask</code>	24
<code>==.fastai.torch_core.TensorImage</code>	25
<code>==.fastai.torch_core.TensorMask</code>	25
<code>==.torch.Tensor</code>	26
<code>&gt;.fastai.torch_core.TensorMask</code>	26
<code>&gt;=.fastai.torch_core.TensorMask</code>	27
<code>abs</code>	27
<code>abs.fastai.torch_core.TensorMask</code>	28
<code>AccumMetric</code>	28
<code>accuracy</code>	29
<code>accuracy_multi</code>	30
<code>accuracy_thresh_expand</code>	30
<code>Adam</code>	31
<code>adam_step</code>	31
<code>AdaptiveAvgPool</code>	32
<code>AdaptiveConcatPool1d</code>	32
<code>AdaptiveConcatPool2d</code>	33
<code>AdaptiveGANSwitcher</code>	33
<code>AdaptiveLoss</code>	34
<code>adaptive_pool</code>	34
<code>add</code>	35
<code>AddChannels</code>	35
<code>AddNoise</code>	36
<code>add_cyclic_datepart</code>	36
<code>add_datepart</code>	37
<code>AffineCoordTfm</code>	37
<code>affine_coord</code>	38
<code>affine_mat</code>	39
<code>alexnet</code>	39
<code>apply_perspective</code>	40
<code>APScoreBinary</code>	40
<code>APScoreMulti</code>	41
<code>aspect</code>	42
<code>as_array</code>	42
<code>AudioBlock</code>	43
<code>AudioBlock_from_folder</code>	43
<code>AudioGetter</code>	44
<code>AudioPadType</code>	45
<code>AudioSpectrogram</code>	45
<code>AudioTensor</code>	45
<code>AudioTensor_create</code>	46
<code>AudioToMFCC</code>	47
<code>AudioToMFCC_from_cfg</code>	47
<code>AudioToSpec_from_cfg</code>	48
<code>audio_extensions</code>	48
<code>aug_transforms</code>	49

AutoConfig	50
average_grad	51
average_sqr_grad	51
AvgLoss	52
AvgPool	52
AvgSmoothLoss	53
AWD_LSTM	53
awd_lstm_clas_split	54
awd_lstm_lm_split	55
AWD_QRNN	55
BalancedAccuracy	56
BaseLoss	57
BaseTokenizer	57
BasicMelSpectrogram	58
BasicMFCC	59
BasicSpectrogram	60
basic_critic	61
basic_generator	61
BatchNorm	62
BatchNorm1dFlat	63
BBoxBlock	63
BBoxLabeler	64
BBoxLblBlock	64
bb_pad	65
BCELossFlat	66
BCEWithLogitsLossFlat	66
blurr	67
BrierScore	67
BrierScoreMulti	68
bs_find	68
bs_finder	69
bt	69
calculate_rouge	70
Callback	70
Cat	71
catalyst	71
catalyst_model	72
Categorify	72
CategoryBlock	73
ceiling.fastai.torch_core.TensorMask	73
ceiling_	74
ChangeVolume	74
children_and_parameters	75
ClassificationInterpretation_from_learner	75
clean_raw_keys	76
clip_remove_empty	76
cm	77
cnn_config	77

cnn_learner	78
COCOMetric	80
COCOMetricType	81
CohenKappa	81
collab	82
CollabDataLoaders_from_dblock	82
CollabDataLoaders_from_df	83
collab_learner	84
CollectDataCallback	86
colors	86
ColReader	87
ColSplitter	87
combined_flat_anneal	88
competitions_list	88
competition_download_file	89
competition_download_files	90
competition_leaderboard_download	91
competition_list_files	91
competition_submit	92
Contrast	92
ConvLayer	93
convT_norm_relu	94
conv_norm_lr	95
CorpusBLEUMetric	96
cos.fastai.torch_core.TensorMask	96
cosh.fastai.torch_core.TensorMask	97
cosh_	97
cos_	98
crap	98
crappifier	99
create_body	99
create_cnn_model	100
create_fcn	101
create_head	102
create_inception	103
create_mlp	103
create_resnet	104
create_unet_model	105
CropPad	106
CropTime	106
CrossEntropyLossFlat	107
CSVLogger	107
CudaCallback	108
custom_loss	109
CutMix	109
cutout_gaussian	110
CycleGAN	110
CycleGANLoss	111

CycleGANTrainer . . . . .	112
cycle_learner . . . . .	112
DataBlock . . . . .	113
dataloaders . . . . .	114
Datasets . . . . .	115
Data_Loaders . . . . .	116
dcmread . . . . .	117
debias . . . . .	117
Debugger . . . . .	118
decision_plot . . . . .	118
decode_spec_tokens . . . . .	119
default_split . . . . .	119
Delta . . . . .	120
denormalize_imagenet . . . . .	120
densenet121 . . . . .	121
densenet161 . . . . .	121
densenet169 . . . . .	122
densenet201 . . . . .	122
DenseResBlock . . . . .	123
dependence_plot . . . . .	124
DeterministicDihedral . . . . .	125
DeterministicDraw . . . . .	125
DeterministicFlip . . . . .	126
detuplify_pg . . . . .	126
Dice . . . . .	127
Dicom . . . . .	127
dicom_windows . . . . .	127
Dihedral . . . . .	128
DihedralItem . . . . .	129
dihedral_mat . . . . .	129
dim . . . . .	130
dim.fastai.torch_core.TensorMask . . . . .	130
discriminator . . . . .	131
div . . . . .	131
DownmixMono . . . . .	132
dropout_mask . . . . .	132
dummy_eval . . . . .	133
DynamicUnet . . . . .	133
EarlyStoppingCallback . . . . .	134
efficientdet_infer_dl . . . . .	135
efficientdet_learner . . . . .	135
efficientdet_model . . . . .	136
efficientdet_predict_dl . . . . .	136
efficientdet_train_dl . . . . .	137
efficientdet_valid_dl . . . . .	137
Embedding . . . . .	138
EmbeddingDropout . . . . .	138
emb_sz_rule . . . . .	139

error_rate	139
exp	140
exp.fastai.torch_core.TensorMask	140
ExplainedVariance	141
expml	141
expml.fastai.torch_core.TensorMask	142
export_generator	142
exp_rmspe	143
F1Score	143
F1ScoreMulti	144
fastai_version	145
fastaudio	145
faster_rcnn_infer_dl	145
faster_rcnn_learner	146
faster_rcnn_model	147
faster_rcnn_predict_dl	147
faster_rcnn_train_dl	148
faster_rcnn_valid_dl	149
fastinf	149
fa_collate	150
fa_convert	150
FBeta	151
FBetaMulti	151
FetchPredsCallback	152
FileSplitter	153
FillMissing	153
FillStrategy_COMMON	154
FillStrategy_CONSTANT	155
FillStrategy_MEDIAN	155
find_coeffs	155
fine_tune	156
fit.fastai.learner.Learner	157
fit.fastai.tabular.learner.TabularLearner	157
fit.fastai.vision.gan.GANLearner	158
fit_flat_cos	158
fit_flat_lin	159
fit_one_cycle	160
fit_sgdr	161
FixedGANSwitcher	162
fix_fit	162
fix_html	163
Flatten	163
flatten_check	164
flatten_model	164
Flip	165
FlipItem	165
flip_mat	166
float	166

floor.fastai.torch_core.TensorMask . . . . .	167
floor_ . . . . .	167
floor_div . . . . .	168
floor_mod . . . . .	168
fmodule . . . . .	169
FolderDataset . . . . .	169
force_plot . . . . .	170
foreground_acc . . . . .	170
ForgetMultGPU . . . . .	171
forget_mult_CPU . . . . .	171
freeze . . . . .	172
FuncSplitter . . . . .	172
fView . . . . .	173
GANDiscriminativeLR . . . . .	173
GANLearner_from_learners . . . . .	174
GANLearner_wgan . . . . .	175
GANLoss . . . . .	177
GANModule . . . . .	177
GANTrainer . . . . .	178
gan_critic . . . . .	178
gan_loss_from_func . . . . .	179
GatherPredsCallback . . . . .	179
gauss_blur2d . . . . .	180
generate_noise . . . . .	180
get_annotations . . . . .	181
get_audio_files . . . . .	182
get_bias . . . . .	182
get_c . . . . .	183
get_confusion_matrix . . . . .	184
get_data_loaders . . . . .	184
get_dcm_matrix . . . . .	185
get_dicom_files . . . . .	186
get_dls . . . . .	186
get_emb_sz . . . . .	187
get_files . . . . .	188
get_grid . . . . .	189
get_hf_objects . . . . .	190
get_image_files . . . . .	190
get_language_model . . . . .	191
get_preds_cyclegan . . . . .	192
get_text_classifier . . . . .	192
get_text_files . . . . .	193
get_weights . . . . .	194
GradientAccumulation . . . . .	195
GrandparentSplitter . . . . .	195
grayscale . . . . .	196
greater . . . . .	196
greater_or_equal . . . . .	197

HammingLoss . . . . .	197
HammingLossMulti . . . . .	198
has_params . . . . .	198
has_pool_type . . . . .	199
helper . . . . .	199
HF_ARCHITECTURES . . . . .	199
HF_BaseInput . . . . .	200
HF_BaseModelCallback . . . . .	200
HF_BaseModelWrapper . . . . .	201
HF_BeforeBatchTransform . . . . .	201
HF_CausalLMBeforeBatchTransform . . . . .	202
HF_load_dataset . . . . .	203
HF_QABatchTransform . . . . .	205
HF_QABeforeBatchTransform . . . . .	206
HF_QstAndAnsModelCallback . . . . .	207
HF_QuestionAnswerInput . . . . .	207
hf_splitter . . . . .	208
HF_SummarizationBeforeBatchTransform . . . . .	208
HF_SummarizationInput . . . . .	209
HF_SummarizationModelCallback . . . . .	210
HF_TASKS_ALL . . . . .	210
HF_TASKS_AUTO . . . . .	211
HF_Text2TextAfterBatchTransform . . . . .	211
HF_Text2TextBlock . . . . .	212
HF_TextBlock . . . . .	212
HF_TokenCategorize . . . . .	213
HF_TokenCategoryBlock . . . . .	213
HF_TokenClassBeforeBatchTransform . . . . .	214
HF_TokenClassInput . . . . .	215
HF_TokenTensorCategory . . . . .	215
Hook . . . . .	215
HookCallback . . . . .	216
Hooks . . . . .	217
hook_output . . . . .	218
hook_outputs . . . . .	218
hsv2rgb . . . . .	219
Hue . . . . .	219
hug . . . . .	220
icevision . . . . .	220
icevision_Adapter . . . . .	220
icevision_aug_tfms . . . . .	221
icevision_BasicIAATransform . . . . .	222
icevision_BasicTransform . . . . .	222
icevision_Blur . . . . .	223
icevision_ChannelDropout . . . . .	223
icevision_ChannelShuffle . . . . .	224
icevision_CLAHE . . . . .	225
icevision_ClassMap . . . . .	225



icevision_CoarseDropout . . . . .	226
icevision_ColorJitter . . . . .	227
icevision_Compose . . . . .	228
icevision_Crop . . . . .	229
icevision_CropNonEmptyMaskIfExists . . . . .	229
icevision_Cutout . . . . .	230
icevision_Dataset . . . . .	231
icevision_Dataset_from_images . . . . .	232
icevision_Downscale . . . . .	233
icevision_DualIAATransform . . . . .	234
icevision_DualTransform . . . . .	234
icevision_ElasticTransform . . . . .	235
icevision_Equalize . . . . .	236
icevision_FancyPCA . . . . .	237
icevision_FDA . . . . .	238
icevision_FixedSplitter . . . . .	239
icevision_Flip . . . . .	239
icevision_FromFloat . . . . .	240
icevision_GaussianBlur . . . . .	241
icevision_GaussNoise . . . . .	242
icevision_GlassBlur . . . . .	242
icevision_GridDistortion . . . . .	243
icevision_GridDropout . . . . .	245
icevision_HistogramMatching . . . . .	246
icevision_HorizontalFlip . . . . .	247
icevision_HueSaturationValue . . . . .	248
icevision_IAAAdditiveGaussianNoise . . . . .	249
icevision_IAAAffine . . . . .	249
icevision_IAACropAndPad . . . . .	251
icevision_IAAEmboss . . . . .	251
icevision_IAAFliplr . . . . .	252
icevision_IAAFlipud . . . . .	253
icevision_IAAPerspective . . . . .	253
icevision_IAAPiecewiseAffine . . . . .	254
icevision_IAASharpn . . . . .	255
icevision_IAASuperpixels . . . . .	256
icevision_ImageCompression . . . . .	257
icevision_ImageOnlyIAATransform . . . . .	258
icevision_ImageOnlyTransform . . . . .	258
icevision_InvertImg . . . . .	259
icevision_ISONoise . . . . .	259
icevision_JpegCompression . . . . .	260
icevision_LongestMaxSize . . . . .	261
icevision_MaskDropout . . . . .	262
icevision_MedianBlur . . . . .	263
icevision_MotionBlur . . . . .	263
icevision_MultiplicativeNoise . . . . .	264
icevision_Normalize . . . . .	265

icevision_OpticalDistortion . . . . .	266
icevision_PadIfNeeded . . . . .	267
icevision_parse . . . . .	268
icevision_Posterize . . . . .	269
icevision_RandomBrightnessContrast . . . . .	269
icevision_RandomContrast . . . . .	270
icevision_RandomCrop . . . . .	271
icevision_RandomCropNearBBox . . . . .	272
icevision_RandomFog . . . . .	272
icevision_RandomGamma . . . . .	273
icevision_RandomGridShuffle . . . . .	274
icevision_RandomRain . . . . .	275
icevision_RandomResizedCrop . . . . .	276
icevision_RandomRotate90 . . . . .	277
icevision_RandomScale . . . . .	277
icevision_RandomShadow . . . . .	278
icevision_RandomSizedBBoxSafeCrop . . . . .	279
icevision_RandomSizedCrop . . . . .	280
icevision_RandomSnow . . . . .	281
icevision_RandomSplitter . . . . .	282
icevision_RandomSunFlare . . . . .	283
icevision_read_bgr_image . . . . .	284
icevision_read_rgb_image . . . . .	284
icevision_Resize . . . . .	285
icevision_resize_and_pad . . . . .	285
icevision_RGBShift . . . . .	286
icevision_Rotate . . . . .	287
icevision_ShiftScaleRotate . . . . .	288
icevision_SingleSplitSplitter . . . . .	289
icevision_SmallestMaxSize . . . . .	290
icevision_Solarize . . . . .	291
icevision_ToFloat . . . . .	291
icevision_ToGray . . . . .	292
icevision_ToSepia . . . . .	293
icevision_Transpose . . . . .	293
icevision_VerticalFlip . . . . .	294
icnr_init . . . . .	295
IDMap . . . . .	295
Image . . . . .	296
image2tensor . . . . .	296
ImageBlock . . . . .	297
ImageBW_create . . . . .	297
ImageDataLoaders_from_csv . . . . .	298
ImageDataLoaders_from_dblock . . . . .	299
ImageDataLoaders_from_df . . . . .	300
ImageDataLoaders_from_folder . . . . .	301
ImageDataLoaders_from_lists . . . . .	302
ImageDataLoaders_from_name_re . . . . .	303

ImageDataLoaders_from_path_func . . . . .	305
ImageDataLoaders_from_path_re . . . . .	306
imagenet_stats . . . . .	307
Image_create . . . . .	307
Image_open . . . . .	308
Image_resize . . . . .	308
InceptionModule . . . . .	309
IndexSplitter . . . . .	309
init . . . . .	310
init_default . . . . .	310
init_linear . . . . .	311
install_fastai . . . . .	311
InstanceNorm . . . . .	312
IntToFloatTensor . . . . .	313
InvisibleTensor . . . . .	313
in_channels . . . . .	314
is_rmarkdown . . . . .	314
Jaccard . . . . .	315
JaccardCoeff . . . . .	315
JaccardMulti . . . . .	316
kg . . . . .	316
L . . . . .	317
L1LossFlat . . . . .	317
l2_reg . . . . .	318
LabeledBBox . . . . .	319
LabelSmoothingCrossEntropy . . . . .	319
LabelSmoothingCrossEntropyFlat . . . . .	320
Lamb . . . . .	320
Lambda . . . . .	321
lamb_step . . . . .	321
language_model_learner . . . . .	322
Larc . . . . .	323
larc_layer_lr . . . . .	324
larc_step . . . . .	324
layer_info . . . . .	325
Learner . . . . .	325
length . . . . .	326
length.fastai.torch_core.TensorMask . . . . .	326
less . . . . .	327
less_or_equal . . . . .	327
LightingTfm . . . . .	328
LinBnDrop . . . . .	328
LinearDecoder . . . . .	329
LitModel . . . . .	329
LMDataLoader . . . . .	330
LMLearner . . . . .	331
LMLearner_predict . . . . .	332
loaders . . . . .	333

load_dataset . . . . .	334
load_ignore_keys . . . . .	334
load_image . . . . .	335
load_learner . . . . .	335
load_model_text . . . . .	336
load_pre_models . . . . .	336
load_tokenized_csv . . . . .	337
log . . . . .	337
log.fastai.torch_core.TensorMask . . . . .	338
log1p . . . . .	338
log1p.fastai.torch_core.TensorMask . . . . .	339
logical_and . . . . .	339
logical_not . . . . .	340
logical_or . . . . .	340
login . . . . .	341
Lookahead . . . . .	341
LossMetric . . . . .	342
lr_find . . . . .	342
mae . . . . .	343
make_vocab . . . . .	343
mask2bbox . . . . .	344
MaskBlock . . . . .	344
masked_concat_pool . . . . .	345
MaskFreq . . . . .	345
MaskTime . . . . .	346
Mask_create . . . . .	346
mask_from_blur . . . . .	347
mask_rcnn_infer_dl . . . . .	347
mask_rcnn_learner . . . . .	348
mask_rcnn_model . . . . .	348
mask_rcnn_predict_dl . . . . .	349
mask_rcnn_train_dl . . . . .	350
mask_rcnn_valid_dl . . . . .	350
mask_tensor . . . . .	351
match_embeds . . . . .	351
MatthewsCorrCoef . . . . .	352
MatthewsCorrCoefMulti . . . . .	352
max . . . . .	353
max.fastai.torch_core.TensorMask . . . . .	353
MaxPool . . . . .	354
maybe_unsqueeze . . . . .	354
MCDropoutCallback . . . . .	355
mean.fastai.torch_core.TensorMask . . . . .	355
mean.torch.Tensor . . . . .	356
medical . . . . .	356
MergeLayer . . . . .	357
metrics . . . . .	357
migrating_ignite . . . . .	357

migrating_lightning	358
migrating_pytorch	358
min	358
min.fastai.torch_core.TensorMask	359
mish	359
MishJitAutoFn	360
Mish_	360
MixHandler	361
MixUp	361
ModelResetter	362
model_sizes	362
Module	363
Module_test	363
momentum_step	363
most_confused	364
mse	364
MSELossFlat	365
msle	366
MultiCategorize	366
MultiCategoryBlock	367
multiplygit add -A && git commit -m 'staging all files'	367
MultiTargetLoss	368
narrow	368
Net	369
nn	369
nn_loss	370
nn_module	370
NoiseColor	371
NoneReduce	371
noop	372
Normalize	372
NormalizeTS	373
Normalize_from_stats	373
norm_apply_denorm	374
not_equal_to	374
not_equal_to_mask_	375
not__mask	375
Numericalize	376
num_features_model	376
n_px	377
OldRandomCrop	377
one_batch	378
OpenAudio	378
Optimizer	379
OptimWrapper	379
optim_metric	380
or_mask	380
os	381

os_environ_tpu . . . . .	381
pad_conv_norm_relu . . . . .	382
pad_input . . . . .	383
pad_input_chunk . . . . .	383
parallel . . . . .	384
parallel_tokenize . . . . .	384
params . . . . .	385
ParamScheduler . . . . .	385
parent_label . . . . .	386
parsers_AreasMixin . . . . .	386
parsers_BBoxesMixin . . . . .	387
parsers_FasterRCNN . . . . .	387
parsers_FilepathMixin . . . . .	388
parsers_ImageidMixin . . . . .	388
parsers_IsCrowdsMixin . . . . .	389
parsers_LabelsMixin . . . . .	389
parsers_MaskRCNN . . . . .	390
parsers_MasksMixin . . . . .	390
parsers_SizeMixin . . . . .	391
parsers_voc . . . . .	391
partial . . . . .	392
PartialDL . . . . .	392
PartialLambda . . . . .	393
pca . . . . .	394
PearsonCorrCoef . . . . .	394
Perplexity . . . . .	395
Pipeline . . . . .	396
PixelShuffle_ICNR . . . . .	396
plot . . . . .	397
plot_bs_find . . . . .	397
plot_confusion_matrix . . . . .	398
plot_loss . . . . .	399
plot_lr_find . . . . .	399
plot_top_losses . . . . .	400
PointBlock . . . . .	401
PointScaler . . . . .	401
PooledSelfAttention2d . . . . .	402
PoolFlatten . . . . .	402
PoolingLinearClassifier . . . . .	403
pow . . . . .	403
Precision . . . . .	404
PrecisionMulti . . . . .	404
predict.fastai.learner.Learner . . . . .	405
predict.fastai.tabular.learner.TabularLearner . . . . .	406
preplexity . . . . .	406
PreprocessAudio . . . . .	407
preprocess_audio_folder . . . . .	407
pre_process_squad . . . . .	408

print.fastai.learner.Learner . . . . .	409
print.fastai.tabular.learner.TabularLearner . . . . .	409
print.pydicom.dataset.FileDataset . . . . .	410
python_path . . . . .	410
py_apply . . . . .	411
QHAdam . . . . .	411
qhadam_step . . . . .	412
QRNN . . . . .	412
QRNNLayer . . . . .	413
R2Score . . . . .	414
RAdam . . . . .	415
radam_step . . . . .	415
RandomCrop . . . . .	416
RandomErasing . . . . .	416
RandomResizedCrop . . . . .	417
RandomResizedCropGPU . . . . .	417
RandomSplitter . . . . .	418
RandPair . . . . .	418
RandTransform . . . . .	419
ranger . . . . .	419
RatioResize . . . . .	420
ReadTSBatch . . . . .	421
Recall . . . . .	421
RecallMulti . . . . .	422
ReduceLRonPlateau . . . . .	422
RegressionBlock . . . . .	423
RemoveSilence . . . . .	424
RemoveType . . . . .	424
replace_all_caps . . . . .	425
replace_maj . . . . .	425
replace_rep . . . . .	426
replace_wrep . . . . .	426
Resample . . . . .	427
ResBlock . . . . .	427
reshape . . . . .	429
Resize . . . . .	429
ResizeBatch . . . . .	430
ResizeSignal . . . . .	430
resize_max . . . . .	431
ResNet . . . . .	431
resnet101 . . . . .	432
resnet152 . . . . .	433
resnet18 . . . . .	433
resnet34 . . . . .	434
resnet50 . . . . .	434
ResnetBlock . . . . .	435
resnet_generator . . . . .	435
res_block_1d . . . . .	436

RetinaNet	437
RetinaNetFocalLoss	437
retinanet_	438
reverse_text	438
rgb2hsv	439
rmse	439
RMSProp	440
rms_prop_step	440
rm_useless_spaces	441
RNNDropout	442
RNNRegularizer	442
RocAuc	443
RocAucBinary	443
RocAucMulti	444
Rotate	445
rotate_mat	446
round	446
round.fastai.torch_core.TensorMask	447
Saturation	447
SaveModelCallback	448
SchedCos	448
SchedExp	449
SchedLin	449
SchedNo	450
SchedPoly	450
SEBlock	451
SegmentationDataLoaders_from_label_func	451
SelfAttention	452
SEModule	453
SentenceEncoder	453
SentencePieceTokenizer	454
SeparableBlock	455
sequential	455
SequentialEx	456
SequentialRNN	456
SEResNeXtBlock	457
setup_aug_tfms	457
set_freeze_model	458
set_item_pg	458
SGD	459
sgd_step	459
SGRoll	460
shap	461
shape	461
ShapInterpretation	462
Shortcut	463
ShortEpochCallback	463
show	464



ShowCycleGANImgsCallback	464
ShowGraphCallback	465
show_array	465
show_batch	466
show_image	467
show_images	468
show_preds	469
show_results	470
show_samples	470
sigmoid	471
SigmoidRange	472
sigmoid_	472
sigmoid_range	473
SignalCutout	473
SignalLoss	474
SignalShifter	474
SimpleCNN	475
SimpleSelfAttention	475
sin.fastai.torch_core.TensorMask	476
sinh.fastai.torch_core.TensorMask	476
sin_	477
skm_to_fastai	477
slice	478
sort	478
sort.fastai.torch_core.TensorMask	479
SortedDL	479
SpacyTokenizer	480
SpearmanCorrCoef	481
SpectrogramTransformer	482
spec_add_spaces	482
sqrd	483
sqrt.fastai.torch_core.TensorMask	483
SqueezeNet	484
squeezenet1_0	484
squeezenet1_1	485
stack_train_valid	486
step_stat	486
sub	487
subplots	487
sub_mask	488
summarization_splitter	488
summary.fastai.learner.Learner	489
summary.fastai.tabular.learner.TabularLearner	489
summary_plot	490
swish	490
Swish_	491
tabular	491
TabularDataTable	492

TabularModel	493
TabularTS	494
TabularTSDataloader	495
tabular_config	496
tabular_learner	497
tar_extract_at_filename	498
tensor	499
TensorBBox	499
TensorBBox_create	500
TensorImage	500
TensorImageBW	501
TensorMultiCategory	501
TensorPoint	502
TensorPoint_create	502
TerminateOnNaNCallback	503
test_loader	503
text	504
TextBlock	504
TextBlock_from_df	505
TextBlock_from_folder	506
TextDataLoaders_from_csv	507
TextDataLoaders_from_df	508
TextDataLoaders_from_folder	510
TextLearner	511
TextLearner_load_encoder	512
TextLearner_load_pretrained	513
TextLearner_save_encoder	513
text_classifier_learner	514
TfmdDL	515
TfmdLists	516
TfmResize	517
timm	517
timm_learner	518
timm_list_models	518
tms	519
tokenize1	519
Tokenizer	520
Tokenizer_from_df	520
TokenizeWithRules	521
tokenize_csv	522
tokenize_df	523
tokenize_files	523
tokenize_folder	524
tokenize_texts	525
top_k_accuracy	526
torch	527
total_params	527
ToTensor	528

to_bytes_format . . . . .	528
to_image . . . . .	529
to_matrix . . . . .	529
to_thumb . . . . .	530
to_xla . . . . .	530
TrackerCallback . . . . .	531
trainable_params . . . . .	531
TrainEvalCallback . . . . .	532
train_loader . . . . .	532
Transform . . . . .	533
TransformBlock . . . . .	533
transformers . . . . .	534
TransformersDropOutput . . . . .	534
TransformersTokenizer . . . . .	535
trunc_normal_ . . . . .	535
TSBlock . . . . .	536
TSDataLoaders_from_dfs . . . . .	536
TSDataTable . . . . .	537
TSeries . . . . .	538
TSeries_create . . . . .	539
UnetBlock . . . . .	539
unet_config . . . . .	541
unet_learner . . . . .	542
unfreeze . . . . .	542
uniform_blur2d . . . . .	543
upit . . . . .	543
URLs_ADULT_SAMPLE . . . . .	544
URLs_AG_NEWS . . . . .	544
URLs_AMAZON_REVIEWSAMAZON_REVIEWS . . . . .	545
URLs_AMAZON_REVIEWS_POLARITY . . . . .	546
URLs_BIWI_HEAD_POSE . . . . .	546
URLs_CALTECH_101 . . . . .	547
URLs_CAMVID . . . . .	547
URLs_CAMVID_TINY . . . . .	548
URLs_CARS . . . . .	548
URLs_CIFAR . . . . .	549
URLs_CIFAR_100 . . . . .	549
URLs_COCO_TINY . . . . .	550
URLs_CUB_200_2011 . . . . .	550
URLs_DBPEDIA . . . . .	551
URLs_DOGS . . . . .	551
URLs_FLOWERS . . . . .	552
URLs_FOOD . . . . .	552
URLs_HORSE_2_ZEBRA . . . . .	553
URLs_HUMAN_NUMBERS . . . . .	553
URLs_IMAGENETTE . . . . .	554
URLs_IMAGENETTE_160 . . . . .	554
URLs_IMAGENETTE_320 . . . . .	555

URLs_IMAGEWOOF	555
URLs_IMAGEWOOF_160	556
URLs_IMAGEWOOF_320	556
URLs_IMDB	557
URLs_IMDB_SAMPLE	557
URLs_LSUN_BEDROOMS	558
URLs_ML_SAMPLE	558
URLs_MNIST	559
URLs_MNIST_SAMPLE	559
URLs_MNIST_TINY	560
URLs_MNIST_VAR_SIZE_TINY	560
URLs_MOVIE_LENS_ML_100k	561
URLs_MT_ENG_FRA	561
URLs_OPENAI_TRANSFORMER	562
URLs_PASCAL_2007	562
URLs_PASCAL_2012	563
URLs_PETS	563
URLs_PLANET_SAMPLE	564
URLs_PLANET_TINY	564
URLs_S3_COCO	565
URLs_S3_IMAGE	565
URLs_S3_IMAGELOC	566
URLs_S3_MODEL	566
URLs_S3_NLP	567
URLs_SIIM_SMALL	567
URLs_SKIN_LESION	568
URLs_SOGOOU_NEWS	568
URLs_SPEAKERS10	569
URLs_SPEECHCOMMANDS	569
URLs_WIKITEXT	570
URLs_WIKITEXT_TINY	570
URLs_WT103_BWD	571
URLs_WT103_FWD	571
URLs_YAHOO_ANSWERS	572
URLs_YELP_REVIEWS	572
URLs_YELP_REVIEWS_POLARITY	573
vgg11_bn	573
vgg13_bn	574
vgg16_bn	574
vgg19_bn	575
vision	575
vleaky_relu	576
Voice	576
wandb	577
WandbCallback	578
Warp	579
waterfall_plot	580
WeightDropout	580

WeightedDL . . . . .	581
weight_decay . . . . .	582
win_abdoment_soft . . . . .	583
win_brain . . . . .	583
win_brain_bone . . . . .	583
win_brain_soft . . . . .	584
win_liver . . . . .	584
win_lungs . . . . .	584
win_mediastinum . . . . .	585
win_spine_bone . . . . .	585
win_spine_soft . . . . .	585
win_stroke . . . . .	586
win_subdural . . . . .	586
xla . . . . .	586
XResNet . . . . .	587
xresnet101 . . . . .	587
xresnet152 . . . . .	588
xresnet18 . . . . .	588
xresnet18_deep . . . . .	589
xresnet18_deeper . . . . .	589
xresnet34 . . . . .	590
xresnet34_deep . . . . .	590
xresnet34_deeper . . . . .	591
xresnet50 . . . . .	591
xresnet50_deep . . . . .	592
xresnet50_deeper . . . . .	592
xresnext101 . . . . .	593
xresnext18 . . . . .	593
xresnext34 . . . . .	594
xresnext50 . . . . .	594
xsenet154 . . . . .	595
xse_resnet101 . . . . .	595
xse_resnet152 . . . . .	596
xse_resnet18 . . . . .	596
xse_resnet34 . . . . .	597
xse_resnet50 . . . . .	597
xse_resnext101 . . . . .	598
xse_resnext18 . . . . .	598
xse_resnext18_deep . . . . .	599
xse_resnext18_deeper . . . . .	599
xse_resnext34 . . . . .	600
xse_resnext34_deep . . . . .	600
xse_resnext34_deeper . . . . .	601
xse_resnext50 . . . . .	601
xse_resnext50_deep . . . . .	602
xse_resnext50_deeper . . . . .	602
zoom . . . . .	603
Zoom_ . . . . .	603

zoom_mat . . . . .	604
&.fastai.torch_core.TensorMask . . . . .	605
%f% . . . . .	605
%%.fastai.torch_core.TensorMask . . . . .	606
%/%.fastai.torch_core.TensorMask . . . . .	606
^.fastai.torch_core.TensorMask . . . . .	607

**Index** **608**

\*.fastai.torch\_core.TensorMask  
*Multiply*

### Description

Multiply

### Usage

```
## S3 method for class 'fastai.torch_core.TensorMask'
a * b
```

### Arguments

a	tensor
b	tensor

### Value

tensor

+.fastai.torch\_core.TensorMask  
*Add*

### Description

Add

### Usage

```
## S3 method for class 'fastai.torch_core.TensorMask'
a + b
```

**Arguments**

a            tensor  
b            tensor

**Value**

tensor

---

*+.torch.nn.modules.container.Sequential*  
*Add layers to Sequential*

---

**Description**

Add layers to Sequential

**Usage**

```
## S3 method for class 'torch.nn.modules.container.Sequential'  
a + b
```

**Arguments**

a            sequential model  
b            layer

**Value**

model

---

*/.fastai.torch\_core.TensorMask*  
*Div*

---

**Description**

Div

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
a / b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

<.fastai.torch\_core.TensorMask  
*Less*

---

**Description**

Less

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
a < b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

<=.fastai.torch\_core.TensorMask  
*Less or equal*

---

**Description**

Less or equal

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
a <= b
```



**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

`==.fastai.torch_core.TensorImage`  
*Equal*

---

**Description**

Equal

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorImage'  
a == b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

`==.fastai.torch_core.TensorMask`  
*Equal*

---

**Description**

Equal

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
a == b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

```
==.torch.Tensor
```

*Equal*

---

**Description**

Equal

**Usage**

```
## S3 method for class 'torch.Tensor'
a == b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

```
>.fastai.torch_core.TensorMask
```

*Greater*

---

**Description**

Greater

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'
a > b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

`>=.fastai.torch_core.TensorMask`  
*Greater or equal*

---

**Description**

Greater or equal

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
a >= b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

abs	<i>Abs</i>
-----	------------

---

**Description**

Abs

**Usage**

```
## S3 method for class 'torch.Tensor'  
abs(x)
```

**Arguments**

x	tensor
---	--------

**Value**

tensor

---

```
abs.fastai.torch_core.TensorMask
    Abs
```

---

**Description**

Abs

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'
abs(x)
```

**Arguments**

x                    tensor, e.g.: tensor(-1:-10)

**Value**

tensor

---

AccumMetric

*AccumMetric*

---

**Description**

Stores predictions and targets on CPU in accumulate to perform final calculations with 'func'.

**Usage**

```
AccumMetric(
  func,
  dim_argmax = NULL,
  activation = "no",
  thresh = NULL,
  to_np = FALSE,
  invert_arg = FALSE,
  flatten = TRUE,
  ...
)
```

**Arguments**

func	function
dim_argmax	dimension argmax
activation	activation
thresh	threshold point
to_np	to matrix or not
invert_arg	invert arguments
flatten	flatten
...	additional arguments to pass

**Value**

None

---

accuracy	<i>Accuracy</i>
----------	-----------------

---

**Description**

Compute accuracy with 'targ' when 'pred' is bs \* n\_classes

**Usage**

```
accuracy(inp, targ, axis = -1)
```

**Arguments**

inp	predictions
targ	targets
axis	axis

**Value**

None

accuracy\_multi      *Accuracy\_multi*

---

**Description**

Compute accuracy when 'inp' and 'targ' are the same size.

**Usage**

```
accuracy_multi(inp, targ, thresh = 0.5, sigmoid = TRUE)
```

**Arguments**

inp	predictions
targ	targets
thresh	threshold point
sigmoid	sigmoid

**Value**

None

---

accuracy\_thresh\_expand  
*Accuracy threshold expand*

---

**Description**

Compute accuracy after expanding 'y\_true' to the size of 'y\_pred'.

**Usage**

```
accuracy_thresh_expand(y_pred, y_true, thresh = 0.5, sigmoid = TRUE)
```

**Arguments**

y_pred	predictions
y_true	actuals
thresh	threshold point
sigmoid	sigmoid function

**Value**

None

---

Adam	<i>Adam</i>
------	-------------

---

**Description**

Adam

**Usage**

Adam(...)

**Arguments**

... parameters to pass

**Value**

None

---

adam_step	<i>Adam_step</i>
-----------	------------------

---

**Description**

Step for Adam with 'lr' on 'p'

**Usage**

adam\_step(p, lr, mom, step, sqr\_mom, grad\_avg, sqr\_avg, eps, ...)

**Arguments**

p	p
lr	learning rate
mom	momentum
step	step
sqr_mom	sqr momentum
grad_avg	grad average
sqr_avg	sqr average
eps	epsilon
...	additional arguments to pass

**Value**

None

AdaptiveAvgPool      *AdaptiveAvgPool*

---

**Description**

nn(\$AdaptiveAvgPool layer for 'ndim')

**Usage**

```
AdaptiveAvgPool(sz = 1, ndim = 2)
```

**Arguments**

sz	size
ndim	dimension size

---

AdaptiveConcatPool1d      *AdaptiveConcatPool1d*

---

**Description**

Layer that concats 'AdaptiveAvgPool1d' and 'AdaptiveMaxPool1d'

**Usage**

```
AdaptiveConcatPool1d(size = NULL)
```

**Arguments**

size	output size
------	-------------

**Value**

None



---

AdaptiveConcatPool2d    *AdaptiveConcatPool2d*

---

**Description**

Layer that concatenates 'AdaptiveAvgPool2d' and 'AdaptiveMaxPool2d'

**Usage**

```
AdaptiveConcatPool2d(size = NULL)
```

**Arguments**

size                    output size

**Value**

None

---

AdaptiveGANSwitcher    *Adaptive GAN Switcher*

---

**Description**

Switcher that goes back to generator/critic when the loss goes below 'gen\_thresh'/'crit\_thresh'.

**Usage**

```
AdaptiveGANSwitcher(gen_thresh = NULL, critic_thresh = NULL)
```

**Arguments**

gen\_thresh            generator threshold  
critic\_thresh        discriminator threshold

**Value**

None

---

AdaptiveLoss

*AdaptiveLoss*

---

**Description**

Expand the 'target' to match the 'output' size before applying 'crit'.

**Usage**

```
AdaptiveLoss(crit)
```

**Arguments**

crit            critic

**Value**

Loss object

---

adaptive\_pool

*Adaptive\_pool*

---

**Description**

Adaptive\_pool

**Usage**

```
adaptive_pool(pool_type)
```

**Arguments**

pool\_type        pooling type

**Value**

Nonee

---

add	<i>Add</i>
-----	------------

---

**Description**

Add

Sinh

**Usage**

```
## S3 method for class 'torch.Tensor'  
a + b
```

```
## S3 method for class 'torch.Tensor'  
sinh(x)
```

**Arguments**

a            tensor

b            tensor

x            tensor

**Value**

tensor

tensor

---

AddChannels	<i>Add Channels</i>
-------------	---------------------

---

**Description**

Add 'n\_dim' channels at the end of the input.

**Usage**

```
AddChannels(n_dim)
```

**Arguments**

n\_dim            number of dimensions

---

AddNoise	<i>Add Noise</i>
----------	------------------

---

**Description**

Adds noise of specified color and level to the audio signal

**Usage**

```
AddNoise(noise_level = 0.05, color = 0)
```

**Arguments**

noise_level	noise level
color	int, color

**Value**

None

---

add_cyclic_datepart	<i>Add cyclic datepart</i>
---------------------	----------------------------

---

**Description**

Helper function that adds trigonometric date/time features to a date in the column 'field\_name' of 'df'.

**Usage**

```
add_cyclic_datepart(
  df,
  field_name,
  prefix = NULL,
  drop = TRUE,
  time = FALSE,
  add_linear = FALSE
)
```

**Arguments**

df	df
field_name	field_name
prefix	prefix
drop	drop
time	time
add_linear	add_linear

**Value**

data frame

---

add_datepart	<i>Add datepart</i>
--------------	---------------------

---

**Description**

Helper function that adds columns relevant to a date in the column 'field\_name' of 'df'.

**Usage**

```
add_datepart(df, field_name, prefix = NULL, drop = TRUE, time = FALSE)
```

**Arguments**

df	df
field_name	field_name
prefix	prefix
drop	drop
time	time

**Value**

data frame

---

AffineCoordTfm	<i>AffineCoordTfm</i>
----------------	-----------------------

---

**Description**

Combine and apply affine and coord transforms

**Usage**

```
AffineCoordTfm(
  aff_fs = NULL,
  coord_fs = NULL,
  size = NULL,
  mode = "bilinear",
  pad_mode = "reflection",
  mode_mask = "nearest",
  align_corners = NULL
)
```

**Arguments**

aff_fs	aff fs
coord_fs	coordinate fs
size	size
mode	mode
pad_mode	padding mode
mode_mask	mode mask
align_corners	align corners

**Value**

None

---

affine_coord	<i>Aaffine_coord</i>
--------------	----------------------

---

**Description**

Aaffine\_coord

**Usage**

```

affine_coord(
    x,
    mat = NULL,
    coord_tfm = NULL,
    sz = NULL,
    mode = "bilinear",
    pad_mode = "reflection",
    align_corners = TRUE,
    ...
)

```

**Arguments**

x	tensor
mat	mat
coord_tfm	coordinate tfm
sz	sz
mode	mode
pad_mode	padding mode
align_corners	align corners
...	additional arguments

**Value**

None

---

`affine_mat`*Affline mat*

---

**Description**

Affline mat

**Usage**`affine_mat(...)`**Arguments**`...` parameters to pass**Value**

None

---

`alexnet`*Alexnet*

---

**Description**

AlexNet model architecture

**Usage**`alexnet(pretrained = FALSE, progress)`**Arguments**`pretrained` pretrained or not  
`progress` to see progress bar or not**Details**

"One weird trick..." <<https://arxiv.org/abs/1404.5997>>

**Value**

model

**Examples**

```
## Not run:

alexnet(pretrained = FALSE, progress = TRUE)

## End(Not run)
```

---

apply_perspective	<i>Apply_perspective</i>
-------------------	--------------------------

---

**Description**

Apply perspective tranform on 'coords' with 'coeffs'

**Usage**

```
apply_perspective(coords, coeffs)
```

**Arguments**

coords	coordinates
coeffs	coefficient

**Value**

None

---

APScoreBinary	<i>APScoreBinary</i>
---------------	----------------------

---

**Description**

Average Precision for single-label binary classification problems

**Usage**

```
APScoreBinary(
  axis = -1,
  average = "macro",
  pos_label = 1,
  sample_weight = NULL
)
```



**Arguments**

axis	axis
average	average
pos_label	pos_label
sample_weight	sample_weight

**Value**

None

---

APScoreMulti

*APScoreMulti*

---

**Description**

Average Precision for multi-label classification problems

**Usage**

```
APScoreMulti(  
    sigmoid = TRUE,  
    average = "macro",  
    pos_label = 1,  
    sample_weight = NULL  
)
```

**Arguments**

sigmoid	sigmoid
average	average
pos_label	pos_label
sample_weight	sample_weight

**Value**

None

---

aspect	<i>Aspect</i>
--------	---------------

---

**Description**

Aspect

**Usage**

aspect(img)

**Arguments**

img            image

**Value**

None

---

as_array	<i>As_array</i>
----------	-----------------

---

**Description**

As\_array

**Usage**

as\_array(tensor)

**Arguments**

tensor            tensor object

**Value**

array

---

AudioBlock

*AudioBlock*

---

### Description

A 'TransformBlock' for audios

### Usage

```
AudioBlock(  
    cache_folder = NULL,  
    sample_rate = 16000,  
    force_mono = TRUE,  
    crop_signal_to = NULL  
)
```

### Arguments

cache_folder	cache folder
sample_rate	sample rate
force_mono	force mono or not
crop_signal_to	int, crop signal

### Value

None

---

AudioBlock\_from\_folder

*AudioBlock from folder*

---

### Description

Build a 'AudioBlock' from a 'path' and caches some intermediary results

### Usage

```
AudioBlock_from_folder(  
    path,  
    sample_rate = 16000,  
    force_mono = TRUE,  
    crop_signal_to = NULL  
)
```

**Arguments**

path	directory, path
sample_rate	sample rate
force_mono	force mono or not
crop_signal_to	int, crop signal

**Value**

None

---

AudioGetter

*AudioGetter*

---

**Description**

Create 'get\_audio\_files' partial function that searches path suffix 'suf'

**Usage**

```
AudioGetter(suf = "", recurse = TRUE, folders = NULL)
```

**Arguments**

suf	suffix
recurse	recursive or not
folders	vector, folders

**Details**

and passes along 'kwargs', only in 'folders', if specified.

**Value**

None

---

AudioPadType	<i>AudioPadType module</i>
--------------	----------------------------

---

**Description**

AudioPadType module

**Usage**

AudioPadType()

**Value**

None

---

AudioSpectrogram	<i>AudioSpectrogram module</i>
------------------	--------------------------------

---

**Description**

AudioSpectrogram module

**Usage**

AudioSpectrogram()

**Value**

None

---

AudioTensor	<i>Audio Tensor</i>
-------------	---------------------

---

**Description**

Semantic torch tensor that represents an audio.

**Usage**

AudioTensor(x, sr = NULL)

**Arguments**

x	tensor
sr	sr

**Value**

tensor

---

AudioTensor\_create     *AudioTensor create*

---

**Description**

Creates audio tensor from file

**Usage**

```
AudioTensor_create(  
    fn,  
    cache_folder = NULL,  
    frame_offset = 0,  
    num_frames = -1,  
    normalize = TRUE,  
    channels_first = TRUE  
)
```

**Arguments**

fn	function
cache_folder	cache folder
frame_offset	offset
num_frames	number of frames
normalize	apply normalization or not
channels_first	channels first/last

**Value**

None

---

AudioToMFCC	<i>AudioToMFCC</i>
-------------	--------------------

---

**Description**

Transform to create MFCC features from audio tensors.

**Usage**

```
AudioToMFCC(
  sample_rate = 16000,
  n_mfcc = 40,
  dct_type = 2,
  norm = "ortho",
  log_mels = FALSE,
  melkwargs = NULL
)
```

**Arguments**

sample_rate	sample rate
n_mfcc	number of mel-frequency cepstral coefficients
dct_type	dct type
norm	normalization type
log_mels	apply log to mels
melkwargs	additional arguments for mels

**Value**

None

---

AudioToMFCC_from_cfg	<i>AudioToMFCC from cfg</i>
----------------------	-----------------------------

---

**Description**

Creates AudioToMFCC from configuration file

**Usage**

```
AudioToMFCC_from_cfg(audio_cfg)
```

**Arguments**

audio_cfg	audio configuration
-----------	---------------------

**Value**

None

---

AudioToSpec\_from\_cfg    *AudioToSpec from cfg*

---

**Description**

Creates AudioToSpec from configuration file

**Usage**

AudioToSpec\_from\_cfg(audio\_cfg)

**Arguments**

audio\_cfg        audio configuration

**Value**

None

---

audio\_extensions        *Audio\_extensions*

---

**Description**

get all allowed audio extensions

**Usage**

audio\_extensions()

**Value**

vector



---

aug_transforms	<i>Augmentation</i>
----------------	---------------------

---

## Description

Utility func to easily create a list of flip, rotate, zoom, warp, lighting transforms.

## Usage

```
aug_transforms(  
    mult = 1,  
    do_flip = TRUE,  
    flip_vert = FALSE,  
    max_rotate = 10,  
    min_zoom = 1,  
    max_zoom = 1.1,  
    max_lighting = 0.2,  
    max_warp = 0.2,  
    p_affine = 0.75,  
    p_lighting = 0.75,  
    xtra_tfms = NULL,  
    size = NULL,  
    mode = "bilinear",  
    pad_mode = "reflection",  
    align_corners = TRUE,  
    batch = FALSE,  
    min_scale = 1  
)
```

## Arguments

mult	ratio
do_flip	to do flip
flip_vert	flip vertical or not
max_rotate	maximum rotation
min_zoom	minimum zoom
max_zoom	maximum zoom
max_lighting	maximum lighting
max_warp	maximum warp
p_affine	probability affine
p_lighting	probability lighting
xtra_tfms	extra transformations
size	size of image

mode	mode
pad_mode	padding mode
align_corners	align_corners
batch	batch size
min_scale	minimum scale

**Value**

None

**Examples**

```
## Not run:

URLs_PETS()

path = 'oxford-iiit-pet'

path_img = 'oxford-iiit-pet/images'
fnames = get_image_files(path_img)

dls = ImageDataLoaders_from_name_re(
  path, fnames, pat='(.)_.jpg$',
  item_tfms=Resize(size = 460), bs = 10,
  batch_tfms=list(aug_transforms(size = 224, min_scale = 0.75),
                  Normalize_from_stats( imagenet_stats() )
)
)

## End(Not run)
```

---

AutoConfig*Auto configuration*

---

**Description**

Auto configuration

**Usage**

AutoConfig()

**Value**

None

---

average_grad	<i>Average_grad</i>
--------------	---------------------

---

**Description**

Keeps track of the avg grads of 'p' in 'state' with 'mom'.

**Usage**

```
average_grad(p, mom, dampening = FALSE, grad_avg = NULL, ...)
```

**Arguments**

p	p
mom	momentum
dampening	dampening
grad_avg	grad average
...	additional args to pass

**Value**

None

---

average_sqr_grad	<i>Average_sqr_grad</i>
------------------	-------------------------

---

**Description**

*Average\_sqr\_grad*

**Usage**

```
average_sqr_grad(p, sqr_mom, dampening = TRUE, sqr_avg = NULL, ...)
```

**Arguments**

p	p
sqr_mom	sqr momentum
dampening	dampening
sqr_avg	sqr average
...	additional args to pass

**Value**

None

---

AvgLoss	<i>AvgLoss</i>
---------	----------------

---

**Description**

Flattens input and output, same as nn\$AvgLoss

**Usage**

```
AvgLoss(...)
```

**Arguments**

... parameters to pass

**Value**

Loss object

---

AvgPool	<i>AvgPool</i>
---------	----------------

---

**Description**

nn\$AvgPool layer for 'ndim'

**Usage**

```
AvgPool(ks = 2, stride = NULL, padding = 0, ndim = 2, ceil_mode = FALSE)
```

**Arguments**

ks	kernel size
stride	the stride of the window. Default value is kernel_size
padding	implicit zero padding to be added on both sides
ndim	dimension number
ceil_mode	when True, will use ceil instead of floor to compute the output shape

**Value**

None

---

AvgSmoothLoss	<i>AvgSmoothLoss</i>
---------------	----------------------

---

**Description**

Smooth average of the losses (exponentially weighted with 'beta')

**Usage**

```
AvgSmoothLoss(beta = 0.98)
```

**Arguments**

beta	beta, defaults to 0.98
------	------------------------

**Value**

Loss object

---

AWD_LSTM	<i>AWD_LSTM</i>
----------	-----------------

---

**Description**

AWD-LSTM inspired by <https://arxiv.org/abs/1708.02182>

**Usage**

```
AWD_LSTM(  
  vocab_sz,  
  emb_sz,  
  n_hid,  
  n_layers,  
  pad_token = 1,  
  hidden_p = 0.2,  
  input_p = 0.6,  
  embed_p = 0.1,  
  weight_p = 0.5,  
  bidir = FALSE  
)
```

**Arguments**

vocab_sz	vocab_sz
emb_sz	emb_sz
n_hid	n_hid
n_layers	n_layers
pad_token	pad_token
hidden_p	hidden_p
input_p	input_p
embed_p	embed_p
weight_p	weight_p
bidir	bidir

**Value**

None

---

awd\_lstm\_clas\_split    *Awd\_lstm\_clas\_split*

---

**Description**

Split a RNN ‘model‘ in groups for differential learning rates.

**Usage**

```
awd_lstm_clas_split(model)
```

**Arguments**

model	model
-------	-------

**Value**

None

---

awd_lstm_lm_split	<i>Awd_lstm_lm_split</i>
-------------------	--------------------------

---

**Description**

Split a RNN ‘model’ in groups for differential learning rates.

**Usage**

```
awd_lstm_lm_split(model)
```

**Arguments**

model	model
-------	-------

**Value**

None

---

AWD_QRNN	<i>AWD_QRNN</i>
----------	-----------------

---

**Description**

Same as an AWD-LSTM, but using QRNNs instead of LSTMs

**Usage**

```
AWD_QRNN(  
  vocab_sz,  
  emb_sz,  
  n_hid,  
  n_layers,  
  pad_token = 1,  
  hidden_p = 0.2,  
  input_p = 0.6,  
  embed_p = 0.1,  
  weight_p = 0.5,  
  bidir = FALSE  
)
```

**Arguments**

vocab_sz	vocab_sz
emb_sz	emb_sz
n_hid	n_hid
n_layers	n_layers
pad_token	pad_token
hidden_p	hidden_p
input_p	input_p
embed_p	embed_p
weight_p	weight_p
bidir	bidir

**Value**

None

---

BalancedAccuracy	<i>BalancedAccuracy</i>
------------------	-------------------------

---

**Description**

Balanced Accuracy for single-label binary classification problems

**Usage**

BalancedAccuracy(axis = -1, sample\_weight = NULL, adjusted = FALSE)

**Arguments**

axis	axis
sample_weight	sample_weight
adjusted	adjusted

**References**

None



---

BaseLoss

*BaseLoss*

---

**Description**

Flattens input and output, same as nn\$BaseLoss

**Usage**

```
BaseLoss(...)
```

**Arguments**

... parameters to pass

**Value**

Loss object

---

BaseTokenizer

*BaseTokenizer*

---

**Description**

Basic tokenizer that just splits on spaces

**Usage**

```
BaseTokenizer(split_char = " ")
```

**Arguments**

split\_char separator

**Value**

None

---

BasicMelSpectrogram    *BasicMelSpectrogram*

---

## Description

BasicMelSpectrogram

## Usage

```
BasicMelSpectrogram(  
    sample_rate = 16000,  
    n_fft = 400,  
    win_length = NULL,  
    hop_length = NULL,  
    f_min = 0,  
    f_max = NULL,  
    pad = 0,  
    n_mels = 128,  
    window_fn = torch()$hann_window,  
    power = 2,  
    normalized = FALSE,  
    kwargs = NULL,  
    mel = TRUE,  
    to_db = TRUE  
)
```

## Arguments

sample_rate	sample rate
n_fft	number of fast fourier transforms
win_length	windowing length
hop_length	hopping length
f_min	minimum frequency
f_max	maximum frequency
pad	padding
n_mels	number of mel-spectrograms
window_fn	window function
power	power
normalized	normalized or not
kwargs	additional arguments
mel	mel-spectrogram or not
to_db	to decibels

**Value**

None

---

`BasicMFCC`*Basic MFCC*

---

**Description**

Basic MFCC

**Usage**

```
BasicMFCC(  
    sample_rate = 16000,  
    n_mfcc = 40,  
    dct_type = 2,  
    norm = "ortho",  
    log_mels = FALSE,  
    melkwargs = NULL  
)
```

**Arguments**

<code>sample_rate</code>	sample rate
<code>n_mfcc</code>	number of mel-frequency cepstral coefficients
<code>dct_type</code>	dct type
<code>norm</code>	normalization type
<code>log_mels</code>	apply log to mels
<code>melkwargs</code>	additional arguments for mels

**Value**

None

---

BasicSpectrogram      *BasicSpectrogram*

---

### Description

BasicSpectrogram

### Usage

```
BasicSpectrogram(  
  n_fft = 400,  
  win_length = NULL,  
  hop_length = NULL,  
  pad = 0,  
  window_fn = torch()$hann_window,  
  power = 2,  
  normalized = FALSE,  
  wkwargs = NULL,  
  mel = FALSE,  
  to_db = TRUE  
)
```

### Arguments

n_fft	number of fast fourier transforms
win_length	windowing length
hop_length	hopping length
pad	padding mode
window_fn	window function
power	power
normalized	normalized or not
wkwargs	additional arguments
mel	mel-spectrogram or not
to_db	to decibels

### Value

None

---

basic_critic	<i>Basic critic</i>
--------------	---------------------

---

**Description**

A basic critic for images 'n\_channels' x 'in\_size' x 'in\_size'.

**Usage**

```
basic_critic(in_size, n_channels, ...)
```

**Arguments**

in_size	input size
n_channels	The number of channels
...	additional parameters to pass

**Value**

None

**Examples**

```
## Not run:  
  
critic = basic_critic(in_size = 64, n_channels = 3, n_extra_layers = 1,  
                    act_cls = partial(nn.LeakyReLU, negative_slope = 0.2))  
  
## End(Not run)
```

---

basic_generator	<i>Basic generator</i>
-----------------	------------------------

---

**Description**

A basic generator from 'in\_sz' to images 'n\_channels' x 'out\_size' x 'out\_size'.

**Usage**

```
basic_generator(out_size, n_channels, ...)
```

**Arguments**

out_size	out_size
n_channels	n_channels
...	additional params to pass

**Value**

generator object

**Examples**

```
## Not run:

generator = basic_generator(out_size = 64, n_channels = 3, n_extra_layers = 1)

## End(Not run)
```

---

BatchNorm

*BatchNorm*


---

**Description**

BatchNorm layer with ‘nf’ features and ‘ndim’ initialized depending on ‘norm\_type’.

**Usage**

```
BatchNorm(
  nf,
  ndim = 2,
  norm_type = 1,
  eps = 1e-05,
  momentum = 0.1,
  affine = TRUE,
  track_running_stats = TRUE
)
```

**Arguments**

nf	input shape
ndim	dimension number
norm_type	normalization type
eps	epsilon
momentum	momentum
affine	affine
track_running_stats	track running statistics

**Value**

None

---

BatchNorm1dFlat	<i>BatchNorm1dFlat</i>
-----------------	------------------------

---

**Description**

'nn.BatchNorm1d', but first flattens leading dimensions

**Usage**

```
BatchNorm1dFlat(
    num_features,
    eps = 1e-05,
    momentum = 0.1,
    affine = TRUE,
    track_running_stats = TRUE
)
```

**Arguments**

num_features	number of features
eps	epsilon
momentum	momentum
affine	affine
track_running_stats	track running statistics

**Value**

None

---

BBoxBlock	<i>BBoxBlock</i>
-----------	------------------

---

**Description**

A 'TransformBlock' for bounding boxes in an image

**Usage**

```
BBoxBlock()
```

**Value**

None

---

 BBoxLabeler

*BBoxLabeler*


---

**Description**

Delegates ('\_\_call\_\_', 'decode', 'setup') to ('encodes', 'decodes', 'setups') if 'split\_idx' matches

**Usage**

```
BBoxLabeler(enc = NULL, dec = NULL, split_idx = NULL, order = NULL)
```

**Arguments**

enc	encoder
dec	decoder
split_idx	split by index
order	order

**Value**

None

---

 BBoxLblBlock

*BBoxLblBlock*


---

**Description**

A 'TransformBlock' for labeled bounding boxes, potentially with 'vocab'

**Usage**

```
BBoxLblBlock(vocab = NULL, add_na = TRUE)
```

**Arguments**

vocab	vocabulary
add_na	add NA

**Value**

None'



**Examples**

```
## Not run:

URLs_COCO_TINY()

c(images, lbl_bbox) %<-% get_annotations('coco_tiny/train.json')
timg = Transform(ImageBW_create)
idx = 49
c(coco_fn, bbox) %<-% list(paste('coco_tiny/train', images[[idx]], sep = '/'),
                          lbl_bbox[[idx]])
coco_img = timg(coco_fn)

tbbox = LabeledBBox(TensorBBox(bbox[[1]]), bbox[[2]])

coco_bb = function(x) {
  TensorBBox_create(bbox[[1]])
}

coco_lbl = function(x) {
  bbox[[2]]
}

coco_dsrc = Datasets(c(rep(coco_fn, 10)),
                    list(Image_create(), list(coco_bb),
                          list(coco_lbl, MultiCategorize(add_na = TRUE) )
                    ), n_inp = 1)

coco_tdl = TfmdDL(coco_dsrc, bs = 9,
                 after_item = list(BBoxLabeler(), PointScaler(),
                                   ToTensor()),
                 after_batch = list(IntToFloatTensor(), aug_transforms())
)

coco_tdl %>% show_batch(dpi = 200)

## End(Not run)
```

---

bb\_pad

*Bb\_pad*


---

**Description**

Function that collect ‘samples’ of labelled bboxes and adds padding with ‘pad\_idx’.

**Usage**

```
bb_pad(samples, pad_idx = 0)
```

**Arguments**

samples	samples
pad_idx	pad index

**Value**

None

---

BCELossFlat	<i>BCELossFlat</i>
-------------	--------------------

---

**Description**

Flattens input and output, same as nn\$BCELoss

**Usage**

```
BCELossFlat(...)
```

**Arguments**

...	parameters to pass
-----	--------------------

**Value**

Loss object

---

BCEWithLogitsLossFlat	<i>BCEWithLogitsLossFlat</i>
-----------------------	------------------------------

---

**Description**

BCEWithLogitsLossFlat

**Usage**

```
BCEWithLogitsLossFlat(...)
```

**Arguments**

...	parameters to pass
-----	--------------------

**Value**

Loss object

---

blurr	<i>Hugging Face module</i>
-------	----------------------------

---

**Description**

Hugging Face module  
Blurr module

**Usage**

```
blurr()

blurr()
```

**Value**

None  
None

---

BrierScore	<i>BrierScore</i>
------------	-------------------

---

**Description**

Brier score for single-label classification problems

**Usage**

```
BrierScore(axis = -1, sample_weight = NULL, pos_label = NULL)
```

**Arguments**

axis	axis
sample_weight	sample_weight
pos_label	pos_label

**Value**

None

---

BrierScoreMulti	<i>BrierScoreMulti</i>
-----------------	------------------------

---

**Description**

Brier score for multi-label classification problems

**Usage**

```
BrierScoreMulti(
  thresh = 0.5,
  sigmoid = TRUE,
  sample_weight = NULL,
  pos_label = NULL
)
```

**Arguments**

thresh	thresh
sigmoid	sigmoid
sample_weight	sample_weight
pos_label	pos_label

**Value**

None

---

bs_find	<i>Bs_find</i>
---------	----------------

---

**Description**

Launch a mock training to find a good batch size to minimize training time.

**Usage**

```
bs_find(
  object,
  lr,
  num_it = NULL,
  n_batch = 5,
  simulate_multi_gpus = TRUE,
  show_plot = TRUE
)
```

**Arguments**

object	model/learner
lr	learning rate
num_it	number of iterations
n_batch	number of batches
simulate_multi_gpus	simulate on multi gpus or not
show_plot	show plot or not

**Details**

However, it may not be a good batch size to minimize the validation loss. A good batch size is where the Simple Noise Scale converge ignoring the small growing trend with the number of iterations if exists. The optimal batch size is about an order the magnitud where Simple Noise scale converge. Typically, the optimal batch size in image classification problems will be 2-3 times lower where

---

bs_finder	<i>Bs finder</i>
-----------	------------------

---

**Description**

Bs finder

**Usage**

bs\_finder()

**Value**

None

---

bt	<i>Builtins module</i>
----	------------------------

---

**Description**

Builtins module

**Usage**

bt()

**Value**

None

calculate\_rouge      *Calculate\_rouge*

---

**Description**

Calculate\_rouge

**Usage**

```
calculate_rouge(  
  predicted_txts,  
  reference_txts,  
  rouge_keys = c("rouge1", "rouge2", "rougeL"),  
  use_stemmer = TRUE  
)
```

**Arguments**

predicted\_txts predicted texts  
reference\_txts reference texts  
rouge\_keys rouge keys  
use\_stemmer use stemmer or not

**Value**

None

---

Callback      *Callback module*

---

**Description**

Callback module

**Usage**

```
Callback()
```

**Value**

None

---

Cat	<i>Cat</i>
-----	------------

---

**Description**

Concatenate layers outputs over a given dim

**Usage**

```
Cat(layers, dim = 1)
```

**Arguments**

layers	layers
dim	dimension size

**Value**

None

---

catalyst	<i>Catalyst module</i>
----------	------------------------

---

**Description**

Catalyst module

**Usage**

```
catalyst()
```

**Value**

None

---

catalyst_model	<i>Catalyst model</i>
----------------	-----------------------

---

**Description**

Catalyst model

**Usage**

```
catalyst_model()
```

**Value**

model

---

Categorify	<i>Categorify</i>
------------	-------------------

---

**Description**

Transform the categorical variables to that type.

**Usage**

```
Categorify(cat_names, cont_names)
```

**Arguments**

cat_names	The names of the categorical variables
cont_names	The names of the continuous variables

**Value**

None



---

CategoryBlock	<i>CategoryBlock</i>
---------------	----------------------

---

**Description**

‘TransformBlock‘ for single-label categorical targets

**Usage**

```
CategoryBlock(vocab = NULL, sort = TRUE, add_na = FALSE)
```

**Arguments**

vocab	vocabulary
sort	sort or not
add_na	add NA

**Value**

Block object

---

ceiling.fastai.torch_core.TensorMask
<i>Ceil</i>

---

**Description**

Ceil

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'
ceiling(x)
```

**Arguments**

x	tensor
---	--------

**Value**

tensor

ceiling\_

*Ceil*

---

**Description**

Ceil

**Usage**

```
## S3 method for class 'torch.Tensor'  
ceiling(x)
```

**Arguments**

x            tensor

**Value**tensor

---

ChangeVolume

*Change Volume*

---

**Description**

Changes the volume of the signal

**Usage**

```
ChangeVolume(p = 0.5, lower = 0.5, upper = 1.5)
```

**Arguments**p            probability  
lower        lower bound  
upper        upper bound**Value**

None

---

 children\_and\_parameters

*Children\_and\_parameters*


---

**Description**

Return the children of 'm' and its direct parameters not registered in modules.

**Usage**

```
children_and_parameters(m)
```

**Arguments**

m	parameters
---	------------

**Value**

None

---

ClassificationInterpretation\_from\_learner

*ClassificationInterpretation\_from\_learner*


---

**Description**

Construct interpretation object from a learner

**Usage**

```
ClassificationInterpretation_from_learner(
  learn,
  ds_idx = 1,
  dl = NULL,
  act = NULL
)
```

**Arguments**

learn	learner/model
ds_idx	ds by index
dl	dataloader
act	activation

**Value**

interpretation object

---

clean_raw_keys	<i>Clean_raw_keys</i>
----------------	-----------------------

---

**Description**

Clean\_raw\_keys

**Usage**

clean\_raw\_keys(wgts)

**Arguments**

wgts	wgts
------	------

**Value**

None

---

clip_remove_empty	<i>Clip_remove_empty</i>
-------------------	--------------------------

---

**Description**

Clip bounding boxes with image border and label background the empty ones

**Usage**

clip\_remove\_empty(bbox, label)

**Arguments**

bbox	bbox
label	label

**Value**

None

---

cm	<i>Cm module</i>
----	------------------

---

**Description**

Cm module

**Usage**

cm()

**Value**

None

---

cnn_config	<i>Cnn config</i>
------------	-------------------

---

**Description**

Convenience function to easily create a config for 'create\_cnn\_model'

**Usage**

```
cnn_config(
  cut = NULL,
  pretrained = TRUE,
  n_in = 3,
  init = nn()$init$kaiming_normal_,
  custom_head = NULL,
  concat_pool = TRUE,
  lin_ftns = NULL,
  ps = 0.5,
  bn_final = FALSE,
  lin_first = FALSE,
  y_range = NULL
)
```

**Arguments**

cut	cut
pretrained	pre-trained or not
n_in	input shape
init	initializer

custom_head	custom head
concat_pool	concatenate pooling
lin_ftrs	linear filters
ps	parameter server
bn_final	batch normalization final
lin_first	linear first
y_range	y_range

**Value**

None

---

cnn_learner	<i>Cnn_learner</i>
-------------	--------------------

---

**Description**

Build a convnet style learner from ‘dls’ and ‘arch’

**Usage**

```

cnn_learner(
  dls,
  arch,
  loss_func = NULL,
  pretrained = TRUE,
  cut = NULL,
  splitter = NULL,
  y_range = NULL,
  config = NULL,
  n_out = NULL,
  normalize = TRUE,
  opt_func = Adam(),
  lr = 0.001,
  cbs = NULL,
  metrics = NULL,
  path = NULL,
  model_dir = "models",
  wd = NULL,
  wd_bn_bias = FALSE,
  train_bn = TRUE,
  moms = list(0.95, 0.85, 0.95)
)

```

**Arguments**

dls	data loader object
arch	a model architecture
loss_func	loss function
pretrained	pre-trained or not
cut	cut
splitter	It is a function that takes self.model and returns a list of parameter groups (or just one parameter group if there are no different parameter groups).
y_range	y_range
config	configuration
n_out	the number of out
normalize	normalize
opt_func	The function used to create the optimizer
lr	learning rate
cbs	Cbs is one or a list of Callbacks to pass to the Learner.
metrics	It is an optional list of metrics, that can be either functions or Metrics.
path	The folder where to work
model_dir	Path and model_dir are used to save and/or load models.
wd	It is the default weight decay used when training the model.
wd_bn_bias	It controls if weight decay is applied to BatchNorm layers and bias.
train_bn	It controls if BatchNorm layers are trained even when they are supposed to be frozen according to the splitter.
moms	The default momentums used in Learner.fit_one_cycle.

**Value**

learner object

**Examples**

```
## Not run:

URLs_MNIST_SAMPLE()
# transformations
tfms = aug_transforms(do_flip = FALSE)
path = 'mnist_sample'
bs = 20

#load into memory
data = ImageDataLoaders_from_folder(path, batch_tfms = tfms, size = 26, bs = bs)

learn = cnn_learner(data, resnet18(), metrics = accuracy, path = getwd())
```

```
## End(Not run)
```

---

COCOMetric

*COCOMetric*

---

## Description

Wrapper around [cocoapi evaluator](<https://github.com/cocodataset/cocoapi>)

## Usage

```
COCOMetric(  
  metric_type = COCOMetricType()$bbox,  
  print_summary = FALSE,  
  show_pbar = FALSE  
)
```

## Arguments

`metric_type`     Dependent on the task you're solving.  
`print_summary`   If 'TRUE', prints a table with statistics.  
`show_pbar`        If 'TRUE' shows pbar when preparing the data for evaluation.

## Details

Calculates average precision. # Arguments `metric_type`: Dependent on the task you're solving. `print_summary`: If 'TRUE', prints a table with statistics. `show_pbar`: If 'TRUE' shows pbar when preparing the data for evaluation.

## Value

None



---

COCOMetricType	<i>COCOMetricType</i>
----------------	-----------------------

---

**Description**

Available options for 'COCOMetric'

**Usage**

COCOMetricType()

**Value**

None

---

CohenKappa	<i>CohenKappa</i>
------------	-------------------

---

**Description**

Cohen kappa for single-label classification problems

**Usage**

CohenKappa(axis = -1, labels = NULL, weights = NULL, sample\_weight = NULL)

**Arguments**

axis	axis
labels	labels
weights	weights
sample_weight	sample_weight

**Value**

None

---

collab	<i>Collab module</i>
--------	----------------------

---

**Description**

Collab module

**Usage**

collab()

**Value**

None

---

CollabDataLoaders_from_dblock	<i>CollabDataLoaders_from_dblock</i>
-------------------------------	--------------------------------------

---

**Description**

Create a dataloaders from a given ‘dblock’

**Usage**

```
CollabDataLoaders_from_dblock(
    dblock,
    source,
    path = ".",
    bs = 64,
    val_bs = NULL,
    shuffle_train = TRUE,
    device = NULL
)
```

**Arguments**

dblock	dblock
source	source
path	The folder where to work
bs	The batch size
val_bs	The batch size for the validation DataLoader (defaults to bs)
shuffle_train	If we shuffle the training DataLoader or not
device	device

**Value**

None

---

CollabDataLoaders\_from\_df  
*CollabDataLoaders\_from\_df*

---

**Description**

Create a 'DataLoaders' suitable for collaborative filtering from 'ratings'.

**Usage**

```
CollabDataLoaders_from_df(
    ratings,
    valid_pct = 0.2,
    user_name = NULL,
    item_name = NULL,
    rating_name = NULL,
    seed = NULL,
    path = ".",
    bs = 64,
    val_bs = NULL,
    shuffle_train = TRUE,
    device = NULL
)
```

**Arguments**

ratings	ratings
valid_pct	The random percentage of the dataset to set aside for validation (with an optional seed)
user_name	The name of the column containing the user (defaults to the first column)
item_name	The name of the column containing the item (defaults to the second column)
rating_name	The name of the column containing the rating (defaults to the third column)
seed	random seed
path	The folder where to work
bs	The batch size
val_bs	The batch size for the validation DataLoader (defaults to bs)
shuffle_train	If we shuffle the training DataLoader or not
device	the device, e.g. cpu, cuda, and etc.

**Value**

None

**Examples**

```
## Not run:

URLs_MOVIE_LENS_ML_100k()
c(user,item,title) %<-% list('userId','movieId','title')
ratings = fread('ml-100k/u.data', col.names = c(user,item,'rating','timestamp'))
movies = fread('ml-100k/u.item', col.names = c(item, 'title', 'date', 'N', 'url',
                                             paste('g',1:19,sep = '')))
rating_movie = ratings[movies[, .SD, .SDcols=c(item,title)], on = item]
dls = CollabDataLoaders_from_df(rating_movie, seed = 42, valid_pct = 0.1, bs = 64,
item_name=title, path='ml-100k')

## End(Not run)
```

---

collab\_learner

*Collab\_learner*


---

**Description**

Create a Learner for collaborative filtering on ‘dls’.

**Usage**

```
collab_learner(
  dls,
  n_factors = 50,
  use_nn = FALSE,
  emb_szs = NULL,
  layers = NULL,
  config = NULL,
  y_range = NULL,
  loss_func = NULL,
  opt_func = Adam(),
  lr = 0.001,
  splitter = trainable_params(),
  cbs = NULL,
  metrics = NULL,
  path = NULL,
  model_dir = "models",
  wd = NULL,
  wd_bn_bias = FALSE,
  train_bn = TRUE,
  moms = list(0.95, 0.85, 0.95)
)
```

**Arguments**

dls	a data loader object
n_factors	The number of factors
use_nn	use_nn
emb_szs	embedding size
layers	list of layers
config	configuration
y_range	y_range
loss_func	It can be any loss function you like. It needs to be one of fastai's if you want to use Learn.predict or Learn.get_preds, or you will have to implement special methods (see more details after the BaseLoss documentation).
opt_func	The function used to create the optimizer
lr	learning rate
splitter	It is a function that takes self.model and returns a list of parameter groups (or just one parameter group if there are no different parameter groups).
cbs	Cbs is one or a list of Callbacks to pass to the Learner.
metrics	It is an optional list of metrics, that can be either functions or Metrics.
path	The folder where to work
model_dir	Path and model_dir are used to save and/or load models.
wd	It is the default weight decay used when training the model.
wd_bn_bias	It controls if weight decay is applied to BatchNorm layers and bias.
train_bn	It controls if BatchNorm layers are trained even when they are supposed to be frozen according to the splitter.
moms	The default momentums used in Learner.fit_one_cycle.

**Value**

learner object

**Examples**

```
## Not run:

URLs_MOVIE_LENS_ML_100k()
c(user,item,title) %<-% list('userId','movieId','title')
ratings = fread('ml-100k/u.data', col.names = c(user,item,'rating','timestamp'))
movies = fread('ml-100k/u.item', col.names = c(item, 'title', 'date', 'N', 'url',
      paste('g',1:19,sep = '')))
rating_movie = ratings[movies[, .SD, .SDcols=c(item,title)], on = item]
dls = CollabDataLoaders_from_df(rating_movie, seed = 42, valid_pct = 0.1, bs = 64,
item_name=title, path='ml-100k')

learn = collab_learner(dls, n_factors = 40, y_range=c(0, 5.5))
```

```
learn %>% fit_one_cycle(1, 5e-3, wd = 1e-1)

## End(Not run)
```

---

CollectDataCallback    *CollectDataCallback*

---

### Description

Collect all batches, along with pred and loss, into self.data. Mainly for testing

### Usage

```
CollectDataCallback(...)

CollectDataCallback(...)
```

### Arguments

...                    arguments to pass

### Value

None  
None

---

colors                    *Colors module*

---

### Description

Colors module

### Usage

```
colors()
```

### Value

None

---

 ColReader

*ColReader*


---

**Description**

Read 'cols' in 'row' with potential 'pref' and 'suff'

**Usage**

```
ColReader(cols, pref = "", suff = "", label_delim = NULL)
```

**Arguments**

cols	columns
pref	pref
suff	suffix
label_delim	label separator

**Value**

None

---

ColSplitter

*ColSplitter*


---

**Description**

Split 'items' (supposed to be a dataframe) by value in 'col'

**Usage**

```
ColSplitter(col = "is_valid")
```

**Arguments**

col	column
-----	--------

**Value**

None

---

combined\_flat\_anneal *Combined\_flat\_anneal*

---

**Description**

Create a schedule with constant learning rate 'start\_lr' for 'pct' proportion of the training, and a 'curve\_type' learning rate (till 'end\_lr') for remaining portion of training.

**Usage**

```
combined_flat_anneal(pct, start_lr, end_lr = 0, curve_type = "linear")
```

**Arguments**

pct	Proportion of training with a constant learning rate.
start_lr	Desired starting learning rate, used for beginning pct of training.
end_lr	Desired end learning rate, training will conclude at this learning rate.
curve_type	Curve type for learning rate annealing. Options are 'linear', 'cosine', and 'exponential'.

---

competitions\_list *Competitions list*

---

**Description**

Competitions list

**Usage**

```
competitions_list(
  group = NULL,
  category = NULL,
  sort_by = NULL,
  page = 1,
  search = NULL
)
```

**Arguments**

group	group to filter result to
category	category to filter result to
sort_by	how to sort the result, see valid_competition_sort_by for options
page	the page to return (default is 1)
search	a search term to use (default is empty string)



**Value**

list of competitions

---

competition\_download\_file

*Competition download file*

---

**Description**

download a competition file to a designated location, or use

**Usage**

```
competition_download_file(  
  competition,  
  file_name,  
  path = NULL,  
  force = FALSE,  
  quiet = FALSE  
)
```

**Arguments**

competition	the name of the competition
file_name	the configuration file name
path	a path to download the file to
force	force the download if the file already exists (default FALSE)
quiet	suppress verbose output (default is FALSE)

**Value**

None

**Examples**

```
## Not run:  
  
com_nm = 'titanic'  
  
titanic_files = competition_list_files(com_nm)  
titanic_files = lapply(1:length(titanic_files),  
                      function(x) as.character(titanic_files[[x]]))  
  
str(titanic_files)  
  
if(!dir.exists(com_nm)) {
```

```
    dir.create(com_nm)
  }

  # download via api
  competition_download_files(competition = com_nm, path = com_nm, unzip = TRUE)

## End(Not run)
```

---

competition\_download\_files  
*Competition download files*

---

## Description

Competition download files

## Usage

```
competition_download_files(  
  competition,  
  path = NULL,  
  force = FALSE,  
  quiet = FALSE,  
  unzip = FALSE  
)
```

## Arguments

competition	the name of the competition
path	a path to download the file to
force	force the download if the file already exists (default FALSE)
quiet	suppress verbose output (default is TRUE)
unzip	unzip downloaded files

## Value

None

---

competition\_leaderboard\_download  
*Competition leaderboard download*

---

### **Description**

Download competition leaderboards

### **Usage**

```
competition_leaderboard_download(competition, path, quiet = TRUE)
```

### **Arguments**

competition	the name of the competition
path	a path to download the file to
quiet	suppress verbose output (default is TRUE)

### **Value**

data frame

---

competition\_list\_files  
*Competition list files*

---

### **Description**

list files for competition

### **Usage**

```
competition_list_files(competition)
```

### **Arguments**

competition	the name of the competition
-------------	-----------------------------

### **Value**

list of files

**Examples**

```
## Not run:

com_nm = 'titanic'
titanic_files = competition_list_files(com_nm)

## End(Not run)
```

---

competition_submit	<i>Competition submit</i>
--------------------	---------------------------

---

**Description**

Competition submit

**Usage**

```
competition_submit(file_name, message, competition, quiet = FALSE)
```

**Arguments**

file_name	the competition metadata file
message	the submission description
competition	the competition name
quiet	suppress verbose output (default is FALSE)

**Value**

None

---

Contrast	<i>Contrast</i>
----------	-----------------

---

**Description**

Apply change in contrast of 'max\_lighting' to batch of images with probability 'p'.

**Usage**

```
Contrast(max_lighting = 0.2, p = 0.75, draw = NULL, batch = FALSE)
```

**Arguments**

max_lighting	maximum lighting
p	probability
draw	draw
batch	batch

**Value**

None

ConvLayer

*ConvLayer***Description**

Create a sequence of convolutional ('ni' to 'nf'), ReLU (if 'use\_activ') and 'norm\_type' layers.

**Usage**

```
ConvLayer(
  ni,
  nf,
  ks = 3,
  stride = 1,
  padding = NULL,
  bias = NULL,
  ndim = 2,
  norm_type = 1,
  bn_1st = TRUE,
  act_cls = nn()$ReLU,
  transpose = FALSE,
  init = "auto",
  xtra = NULL,
  bias_std = 0.01,
  dilation = 1,
  groups = 1,
  padding_mode = "zeros"
)
```

**Arguments**

ni	number of inputs
nf	outputs/ number of features
ks	kernel size
stride	stride

padding	padding
bias	bias
ndim	dimension number
norm_type	normalization type
bn_1st	batch normalization 1st
act_cls	activation
transpose	transpose
init	initializer
xtra	xtra
bias_std	bias standard deviation
dilation	specify the dilation rate to use for dilated convolution
groups	groups size
padding_mode	padding mode, e.g 'zeros'

**Value**

None

---

convT_norm_relu	<i>ConvT_norm_relu</i>
-----------------	------------------------

---

**Description**

ConvT\_norm\_relu

**Usage**

convT\_norm\_relu(ch\_in, ch\_out, norm\_layer, ks = 3, stride = 2, bias = TRUE)

**Arguments**

ch_in	input
ch_out	output
norm_layer	normalziation layer
ks	kernel size
stride	stride size
bias	bias true or not

**Value**

None

---

`conv_norm_lr`*Conv\_norm\_lr*

---

**Description**

Conv\_norm\_lr

**Usage**

```
conv_norm_lr(  
  ch_in,  
  ch_out,  
  norm_layer = NULL,  
  ks = 3,  
  bias = TRUE,  
  pad = 1,  
  stride = 1,  
  activ = TRUE,  
  slope = 0.2,  
  init = nn()$init$normal_,  
  init_gain = 0.02  
)
```

**Arguments**

ch_in	input
ch_out	output
norm_layer	normalziation layer
ks	kernel size
bias	bias
pad	pad
stride	stride
activ	activation
slope	slope
init	initializer
init_gain	initializer gain

**Value**

None

---

CorpusBLEUMetric      *CorpusBLEUMetric*

---

**Description**

Blueprint for defining a metric

**Usage**

```
CorpusBLEUMetric(vocab_sz = 5000, axis = -1)
```

**Arguments**

vocab_sz	vocab_sz
axis	axis

**Value**

None

---

cos.fastai.torch\_core.TensorMask  
*Cos*

---

**Description**

Cos

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
cos(x)
```

**Arguments**

x	tensor
---	--------

**Value**

tensor



---

*cosh.fastai.torch\_core.TensorMask*  
*Cosh*

---

**Description**

Cosh

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
cosh(x)
```

**Arguments**

x                    tensor

**Value**

tensor

---

*cosh\_*                    *Cosh*

---

**Description**

Cosh

**Usage**

```
## S3 method for class 'torch.Tensor'  
cosh(x)
```

**Arguments**

x                    tensor

**Value**

tensor

---

cos_	<i>Cos</i>
------	------------

---

**Description**

Cos

**Usage**

```
## S3 method for class 'torch.Tensor'  
cos(x)
```

**Arguments**

x            tensor

**Value**

tensor

---

crap	<i>Crappify module</i>
------	------------------------

---

**Description**

Crappify module

**Usage**

```
crap()
```

**Value**

None

---

`crappifier`*Crappifier*

---

**Description**

Crappifier

**Usage**`crappifier(path_lr, path_hr)`**Arguments**

<code>path_lr</code>	path from (origin)
<code>path_hr</code>	path to (destination)

**Value**

None

**Examples**

```
## Not run:  
  
items = get_image_files(path_hr)  
parallel(crappifier(path_lr, path_hr), items)  
  
## End(Not run)
```

---

`create_body`*Create\_body*

---

**Description**

Cut off the body of a typically pretrained ‘arch’ as determined by ‘cut’

**Usage**`create_body(...)`**Arguments**

<code>...</code>	parameters to pass
------------------	--------------------

**Value**

None

**Examples**

```
## Not run:

encoder = create_body(resnet34(), pretrained = TRUE)

## End(Not run)
```

---

create_cnn_model	<i>Create_cnn_model</i>
------------------	-------------------------

---

**Description**

Create custom convnet architecture using ‘arch’, ‘n\_in’ and ‘n\_out’

**Usage**

```
create_cnn_model(
  arch,
  n_out,
  cut = NULL,
  pretrained = TRUE,
  n_in = 3,
  init = nn()$init$kaiming_normal_,
  custom_head = NULL,
  concat_pool = TRUE,
  lin_fts = NULL,
  ps = 0.5,
  bn_final = FALSE,
  lin_first = FALSE,
  y_range = NULL
)
```

**Arguments**

arch	a model architecture
n_out	number of outs
cut	cut
pretrained	pretrained model or not
n_in	input shape

init	initializer
custom_head	custom head
concat_pool	concatenate pooling
lin_ftrs	linear filters
ps	parameter server
bn_final	batch normalization final
lin_first	linear first
y_range	y_range

**Value**

None

---

create_fcn	<i>Create_fcn</i>
------------	-------------------

---

**Description**

A bunch of convolutions stacked together.

**Usage**

```
create_fcn(ni, nout, ks = 9, conv_sizes = c(128, 256, 128), stride = 1)
```

**Arguments**

ni	number of input channels
nout	output shape
ks	kernel size
conv_sizes	convolution sizes
stride	stride

**Value**

model

---

create_head	<i>Create_head</i>
-------------	--------------------

---

### Description

Model head that takes 'nf' features, runs through 'lin\_ftrs', and out 'n\_out' classes.

### Usage

```
create_head(  
    nf,  
    n_out,  
    lin_ftrs = NULL,  
    ps = 0.5,  
    concat_pool = TRUE,  
    bn_final = FALSE,  
    lin_first = FALSE,  
    y_range = NULL  
)
```

### Arguments

nf	number of features
n_out	number of out features
lin_ftrs	linear features
ps	parameter server
concat_pool	concatenate pooling
bn_final	batch normalization final
lin_first	linear first
y_range	y_range

### Value

None

---

create_inception	<i>Create_inception</i>
------------------	-------------------------

---

**Description**

Creates an InceptionTime arch from ‘ni’ channels to ‘nout’ outputs.

**Usage**

```
create_inception(
  ni,
  nout,
  kss = c(39, 19, 9),
  depth = 6,
  bottleneck_size = 32,
  nb_filters = 32,
  head = TRUE
)
```

**Arguments**

ni	number of input channels
nout	number of outputs, should be equal to the number of classes for classification tasks.
kss	kernel sizes for the inception Block.
depth	depth
bottleneck_size	The number of channels on the convolution bottleneck.
nb_filters	Channels on the convolution of each kernel.
head	TRUE if we want a head attached.

**Value**

model

---

create_mlp	<i>Create_mlp</i>
------------	-------------------

---

**Description**

A simple model builder to create a bunch of BatchNorm1d, Dropout and Linear layers, with “act\_fn” activations.

**Usage**

```
create_mlp(ni, nout, linear_sizes = c(500, 500, 500))
```

**Arguments**

ni	number of input channels
nout	output shape
linear_sizes	linear output sizes

**Value**

model

---

create_resnet	<i>Create_resnet</i>
---------------	----------------------

---

**Description**

Basic 11 Layer - 1D resnet builder

**Usage**

```
create_resnet(  
  ni,  
  nout,  
  kss = c(9, 5, 3),  
  conv_sizes = c(64, 128, 128),  
  stride = 1  
)
```

**Arguments**

ni	number of input channels
nout	output shape
kss	kernel size
conv_sizes	convolution sizes
stride	stride

**Value**

model



---

create\_unet\_model      *Create\_unet\_model*

---

## Description

Create custom unet architecture

## Usage

```
create_unet_model(
  arch,
  n_out,
  img_size,
  pretrained = TRUE,
  cut = NULL,
  n_in = 3,
  blur = FALSE,
  blur_final = TRUE,
  self_attention = FALSE,
  y_range = NULL,
  last_cross = TRUE,
  bottle = FALSE,
  act_cls = nn()$ReLU,
  init = nn()$init$kaiming_normal_,
  norm_type = NULL
)
```

## Arguments

arch	architecture
n_out	number of out features
img_size	image shape
pretrained	pretrained or not
cut	cut
n_in	number of input
blur	blur is used to avoid checkerboard artifacts at each layer.
blur_final	blur final is specific to the last layer.
self_attention	self_attention determines if we use a self attention layer at the third block before the end.
y_range	If y_range is passed, the last activations go through a sigmoid rescaled to that range.
last_cross	last_cross
bottle	bottle

act_cls	activation
init	initialzier
norm_type	normalization type

**Value**

None

---

CropPad	<i>CropPad</i>
---------	----------------

---

**Description**

Center crop or pad an image to 'size'

**Usage**

CropPad(size, pad\_mode = "zeros", ...)

**Arguments**

size	size
pad_mode	padding mode
...	additional arguments

**Value**

None

---

CropTime	<i>Crop Time</i>
----------	------------------

---

**Description**

Random crops full spectrogram to be length specified in ms by crop\_duration

**Usage**

CropTime(duration, pad\_mode = AudioPadType()\$Zeros)

**Arguments**

duration	int, duration
pad_mode	padding mode, by default 'AudioPadType\$Zeros'

**Value**

None

---

CrossEntropyLossFlat    *CrossEntropyLossFlat*

---

**Description**

Same as ‘nn\$Module’, but no need for subclasses to call ‘super().\_\_init\_\_’

**Usage**

```
CrossEntropyLossFlat(...)
```

**Arguments**

...                    parameters to pass

**Value**

Loss object

---

CSVLogger                    *CSVLogger*

---

**Description**

Basic class handling tweaks of the training loop by changing a ‘Learner’ in various events

**Usage**

```
CSVLogger(fname = "history.csv", append = FALSE)
```

**Arguments**

fname                    file name  
append                    append or not

**Value**

None

## Examples

```
## Not run:

URLs_MNIST_SAMPLE()
# transformations
tfms = aug_transforms(do_flip = FALSE)
path = 'mnist_sample'
bs = 20

#load into memory
data = ImageDataLoaders_from_folder(path, batch_tfms = tfms, size = 26, bs = bs)

learn = cnn_learner(data, resnet18(), metrics = accuracy, path = getwd())

learn %>% fit_one_cycle(2, cbs = CSVLogger())

## End(Not run)
```

---

CudaCallback

*CudaCallback*

---

## Description

Move data to CUDA device

## Usage

```
CudaCallback(device = NULL)
```

## Arguments

device            device name

## Value

None

---

custom_loss	<i>Loss NN module</i>
-------------	-----------------------

---

**Description**

Loss NN module

**Usage**

```
custom_loss()
```

**Value**

None

---

CutMix	<i>CutMix</i>
--------	---------------

---

**Description**

Implementation of '<https://arxiv.org/abs/1905.04899>'

**Usage**

```
CutMix(alpha = 1)
```

**Arguments**

alpha	alpha
-------	-------

**Value**

None

---

cutout_gaussian	<i>Cutout_gaussian</i>
-----------------	------------------------

---

**Description**

Replace all 'areas' in 'x' with  $N(0,1)$  noise

**Usage**

```
cutout_gaussian(x, areas)
```

**Arguments**

x	tensor
areas	areas

**Value**

None

---

CycleGAN	<i>CycleGAN</i>
----------	-----------------

---

**Description**

CycleGAN model.

**Usage**

```
CycleGAN(  
  ch_in = 3,  
  ch_out = 3,  
  n_features = 64,  
  disc_layers = 3,  
  gen_blocks = 9,  
  lsgan = TRUE,  
  drop = 0,  
  norm_layer = NULL  
)
```

**Arguments**

ch_in	input
ch_out	output
n_features	number of features
disc_layers	discriminator layers
gen_blocks	generator blocks
lsgan	ls gan
drop	dropout rate
norm_layer	normalziation layer

**Details**

When called, takes in input batch of real images from both domains and outputs fake images for the opposite domains (with the generators). Also outputs identity images after passing the images into generators that outputs its domain type (needed for identity loss). Attributes: 'G\_A' ('nn.Module'): takes real input B and generates fake input A 'G\_B' ('nn.Module'): takes real input A and generates fake input B 'D\_A' ('nn.Module'): trained to make the difference between real input A and fake input A 'D\_B' ('nn.Module'): trained to make the difference between real input B and fake input B

**Value**

None

---

CycleGANLoss	<i>CycleGANLoss</i>
--------------	---------------------

---

**Description**

CycleGAN loss function. The individual loss terms are also attributes of this class that are accessed by fastai for recording during training.

**Usage**

```
CycleGANLoss(cgan, l_A = 10, l_B = 10, l_idt = 0.5, lsgan = True)
```

**Arguments**

cgan	The CycleGAN model.
l_A	lambda_A, weight of domain A losses. (default=10)
l_B	lambda_B, weight of domain B losses. (default=10)
l_idt	lambda_idt, weight of identity losses. (default=0.5)
lsgan	Whether or not to use LSGAN objective (default=True)

**Details**

Attributes: 'self.cgan' ('nn.Module'): The CycleGAN model. 'self.l\_A' ('float'): lambda\_A, weight of domain A losses. 'self.l\_B' ('float'): lambda\_B, weight of domain B losses. 'self.l\_idt' ('float'): lambda\_idt, weight of identity losses. 'self.crit' ('AdaptiveLoss'): The adversarial loss function (either a BCE or MSE loss depending on 'lsgan' argument) 'self.real\_A' and 'self.real\_B' ('fastai.torch\_core.TensorImage'): Real images from domain A and B. 'self.id\_loss\_A' ('torch.FloatTensor'): The identity loss for domain A calculated in the forward function 'self.id\_loss\_B' ('torch.FloatTensor'): The identity loss for domain B calculated in the forward function 'self.gen\_loss' ('torch.FloatTensor'): The generator loss calculated in the forward function 'self.cyc\_loss' ('torch.FloatTensor'): The cyclic loss calculated in the forward function

---

 CycleGANTrainer

*CycleGANTrainer*


---

**Description**

Learner Callback for training a CycleGAN model.

**Usage**

CycleGANTrainer(...)

**Arguments**

... parameters to pass

**Value**

None

---

 cycle\_learner

*Cycle\_learner*


---

**Description**

Initialize and return a 'Learner' object with the data in 'dls', CycleGAN model 'm', optimizer function 'opt\_func', metrics 'metrics',



**Usage**

```

cycle_learner(
    dls,
    m,
    opt_func = Adam(),
    show_imgs = TRUE,
    imgA = TRUE,
    imgB = TRUE,
    show_img_interval = 10,
    ...
)

```

**Arguments**

dls	dataloader
m	CycleGAN model
opt_func	optimizer
show_imgs	show images
imgA	image a (from)
imgB	image B (to)
show_img_interval	show images interval rafe
...	additional arguments

**Details**

and callbacks ‘cbs’. Additionally, if ‘show\_imgs’ is TRUE, it will show intermediate predictions during training. It will show domain B-to-A predictions if ‘imgA’ is TRUE and/or domain A-to-B predictions if ‘imgB’ is TRUE. Additionally, it will show images every ‘show\_img\_interval’ epochs. ‘Other ‘Learner’ arguments can be passed as well.

**Value**

None

---

DataBlock

*DataBlock*

---

**Description**

Generic container to quickly build ‘Datasets’ and ‘DataLoaders’

**Usage**

```
DataBlock(
    blocks = NULL,
    dl_type = NULL,
    getters = NULL,
    n_inp = NULL,
    item_tfms = NULL,
    batch_tfms = NULL,
    ...
)
```

**Arguments**

blocks	input blocks
dl_type	DL application
getters	how to get dataet
n_inp	n_inp is the number of elements in the tuples that should be considered part of the input and will default to 1 if tfms consists of one set of transforms
item_tfms	One or several transforms applied to the items before batching them
batch_tfms	One or several transforms applied to the batches once they are formed
...	additional parameters to pass

**Value**

Block object

---

dataloaders	<i>Dataloaders from dls object</i>
-------------	------------------------------------

---

**Description**

Create a ‘DataLoaders’ object from ‘source’

**Usage**

```
dataloaders(object, ...)
```

**Arguments**

object	model
...	additional parameters to pass

**Examples**

```
## Not run:

dls = TabularDataTable(df, procs, cat_names, cont_names,
  y_names = dep_var, splits = list(tr_idx, ts_idx) ) %>%
  dataloaders(bs = 50)

## End(Not run)
```

---

 Datasets

*Datasets*


---

**Description**

A dataset that creates a list from each ‘tfms’, passed thru ‘item\_tfms’

**Usage**

```
Datasets(
  items = NULL,
  tfms = NULL,
  tls = NULL,
  n_inp = NULL,
  dl_type = NULL,
  use_list = NULL,
  do_setup = TRUE,
  split_idx = NULL,
  train_setup = TRUE,
  splits = NULL,
  types = NULL,
  verbose = FALSE
)
```

**Arguments**

items	items
tfms	transformations
tls	tls
n_inp	n_inp
dl_type	DL type
use_list	use list
do_setup	do setup
split_idx	split by index

train_setup	train setup
splits	splits
types	types
verbose	verbose

**Value**

None

---

Data\_Loaders
*Data Loaders***Description**

Data Loaders

**Usage**

Data\_Loaders(...)

**Arguments**

... parameters to pass

**Value**

loader object

**Examples**

```
## Not run:

data = Data_Loaders(train_loader, test_loader)

learn = Learner(data, Net(), loss_func = F$nnl_loss,
                opt_func = Adam(), metrics = accuracy, cbs = CudaCallback())

learn %>% fit_one_cycle(1, 1e-2)

## End(Not run)
```

---

dcmread	<i>Read dicom</i>
---------	-------------------

---

**Description**

Open a 'DICOM' file

**Usage**

```
dcmread(fn, force = FALSE)
```

**Arguments**

fn	file name
force	logical, force

**Value**

dicom object

**Examples**

```
## Not run:  
  
img = dcmread('hemorrhage.dcm')  
  
## End(Not run)
```

---

debias	<i>Debias</i>
--------	---------------

---

**Description**

Debias

**Usage**

```
debias(mom, damp, step)
```

**Arguments**

mom	mom
damp	damp
step	step

**Value**

None

---

`Debugger`

---

*Debugger***Description**

A module to debug inside a model

**Usage**`Debugger(...)`**Arguments**

... parameters to pass

**Value**

None

---

`decision_plot`

---

*Decision\_plot***Description**

Visualizes a model's decisions using cumulative SHAP values.

**Usage**`decision_plot(object, class_id = 0, row_idx = -1, dpi = 200, ...)`**Arguments**

<code>object</code>	ShapInterpretation object
<code>class_id</code>	is used to indicate the class of interest for a classification model. It can either be an int or str representation for a class of choice. Each colored line in the plot represents the model's prediction for a single observation.
<code>row_idx</code>	If no index is passed in to use from the data, it will default to the first ten samples on the test set. Note:plotting too many samples at once can make the plot illegible.
<code>dpi</code>	dots per inch
...	additional arguments

**Value**

None

---

decode_spec_tokens	<i>Decode_spec_tokens</i>
--------------------	---------------------------

---

**Description**

Decode the special tokens in 'tokens'

**Usage**

```
decode_spec_tokens(tokens)
```

**Arguments**

tokens	tokens
--------	--------

**Value**

None

---

default_split	<i>Default_split</i>
---------------	----------------------

---

**Description**

Default split of a model between body and head

**Usage**

```
default_split(m)
```

**Arguments**

m	parameters
---	------------

**Value**

None

Delta

*Delta*

---

**Description**

Creates delta with order 1 and 2 from spectrogram and concatenate with the original

**Usage**

```
Delta(width = 9)
```

**Arguments**

```
width          int, width
```

**Value**

None

---

denormalize\_imagenet *Denormalize\_imagenet*

---

**Description**

Denormalize\_imagenet

**Usage**

```
denormalize_imagenet(img)
```

**Arguments**

```
img           img
```

**Value**

None



---

densenet121	<i>Densenet121</i>
-------------	--------------------

---

**Description**

Densenet121

**Usage**

```
densenet121(pretrained = FALSE, progress)
```

**Arguments**

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

"Densely Connected Convolutional Networks" <<https://arxiv.org/pdf/1608.06993.pdf>>

**Value**

model

---

densenet161	<i>Densenet161</i>
-------------	--------------------

---

**Description**

Densenet161

**Usage**

```
densenet161(pretrained = FALSE, progress)
```

**Arguments**

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

"Densely Connected Convolutional Networks" <<https://arxiv.org/pdf/1608.06993.pdf>>

**Value**

model

---

densenet169

*Densenet169*

---

**Description**

Densenet169

**Usage**

```
densenet169(pretrained = FALSE, progress)
```

**Arguments**

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

"Densely Connected Convolutional Networks" <<https://arxiv.org/pdf/1608.06993.pdf>>

**Value**

model

---

densenet201

*Densenet201*

---

**Description**

Densenet201

**Usage**

```
densenet201(pretrained = FALSE, progress)
```

**Arguments**

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

"Densely Connected Convolutional Networks" <<https://arxiv.org/pdf/1608.06993.pdf>>

**Value**

model

---

DenseResBlock	<i>Dense Res Block</i>
---------------	------------------------

---

**Description**

Resnet block of 'nf' features. 'conv\_kwargs' are passed to 'conv\_layer'.

**Usage**

```
DenseResBlock(  
  nf,  
  norm_type = 1,  
  ks = 3,  
  stride = 1,  
  padding = NULL,  
  bias = NULL,  
  ndim = 2,  
  bn_1st = TRUE,  
  act_cls = nn()$ReLU,  
  transpose = FALSE,  
  init = "auto",  
  xtra = NULL,  
  bias_std = 0.01,  
  dilation = 1,  
  groups = 1,  
  padding_mode = "zeros"  
)
```

**Arguments**

nf	number of features
norm_type	normalization type
ks	kernel size
stride	stride
padding	padding
bias	bias
ndim	number of dimensions
bn_1st	batch normalization 1st
act_cls	activation
transpose	transpose
init	initializer
xtra	xtra
bias_std	bias standard deviation

dilation	dilation number
groups	groups number
padding_mode	padding mode

**Value**

block

---

dependence_plot	<i>Dependence_plot</i>
-----------------	------------------------

---

**Description**

Plots the value of a variable on the x-axis and the SHAP value of the same variable on the y-axis. Accepts a class\_id and variable\_name.

**Usage**

```
dependence_plot(object, variable_name = "", class_id = 0, dpi = 200, ...)
```

**Arguments**

object	ShapInterpretation object
variable_name	the name of the column
class_id	is used to indicate the class of interest for a classification model. It can either be an int or str representation for a class of choice. This plot shows how the model depends on the given variable. Vertical dispersion of the datapoints represent interaction effects. Gray ticks along the y-axis are datapoints where the variable's values were NaN.
dpi	dots per inch
...	additional arguments

**Value**

None

---

DeterministicDihedral *DeterministicDihedral*

---

**Description**

Apply a random dihedral transformation to a batch of images with a probability 'p'

**Usage**

```
DeterministicDihedral(  
  size = NULL,  
  mode = "bilinear",  
  pad_mode = "reflection",  
  align_corners = NULL  
)
```

**Arguments**

size	size
mode	mode
pad_mode	padding mode
align_corners	align corners

**Value**

None

---

DeterministicDraw *DeterministicDraw*

---

**Description**

DeterministicDraw

**Usage**

```
DeterministicDraw(vals)
```

**Arguments**

vals	values
------	--------

**Value**

None

---

DeterministicFlip      *DeterministicFlip*

---

**Description**

Flip the batch every other call

**Usage**

```
DeterministicFlip(
  size = NULL,
  mode = "bilinear",
  pad_mode = "reflection",
  align_corners = TRUE,
  ...
)
```

**Arguments**

size	size
mode	mode
pad_mode	padding mode
align_corners	align corners
...	parameters to pass

**Value**

None

---

detuplify\_pg      *Detuplify\_pg*

---

**Description**

Detuplify\_pg

**Usage**

```
detuplify_pg(d)
```

**Arguments**

d	d
---	---

**Value**

None

---

Dice	<i>Dice coefficient</i>
------	-------------------------

---

**Description**

Dice coefficient metric for binary target in segmentation

**Usage**

```
Dice(axis = 1)
```

**Arguments**

axis	axis
------	------

**Value**

None

---

Dicom	<i>Dicom class</i>
-------	--------------------

---

**Description**

Dicom class

**Usage**

```
Dicom()
```

**Value**

None

---

dicom_windows	<i>Dicom_windows module</i>
---------------	-----------------------------

---

**Description**

Dicom\_windows module

**Usage**

```
dicom_windows()
```

**Value**

None

---

Dihedral

*Dihedral*


---

**Description**

Apply a random dihedral transformation to a batch of images with a probability ‘p’

Apply a random dihedral transformation to a batch of images with a probability ‘p’

**Usage**

```
Dihedral(
  p = 0.5,
  draw = NULL,
  size = NULL,
  mode = "bilinear",
  pad_mode = "reflection",
  align_corners = NULL,
  batch = FALSE
)
```

```
Dihedral(
  p = 0.5,
  draw = NULL,
  size = NULL,
  mode = "bilinear",
  pad_mode = "reflection",
  align_corners = NULL,
  batch = FALSE
)
```

**Arguments**

p	probability
draw	draw
size	size
mode	mode
pad_mode	padding mode
align_corners	align corners
batch	batch

**Value**

None

None



---

DihedralItem	<i>DihedralItem</i>
--------------	---------------------

---

**Description**

Randomly flip with probability 'p'

**Usage**

```
DihedralItem(p = 1, nm = NULL, before_call = NULL)
```

**Arguments**

p	probability
nm	nm
before_call	before call

**Value**

None

---

dihedral_mat	<i>Dihedral_mat</i>
--------------	---------------------

---

**Description**

Return a random dihedral matrix

**Usage**

```
dihedral_mat(x, p = 0.5, draw = NULL, batch = FALSE)
```

**Arguments**

x	tensor
p	probability
draw	draw
batch	batch

**Value**

None

dim

*Dim*

---

**Description**

Dim

**Usage**

```
## S3 method for class 'torch.Tensor'  
dim(x)
```

**Arguments**

x            tensor

**Value**tensor

---

dim.fastai.torch\_core.TensorMask

*Dim*

---

**Description**

Dim

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
dim(x)
```

**Arguments**

x            tensor

**Value**

tensor

---

discriminator	<i>Discriminator</i>
---------------	----------------------

---

**Description**

Discriminator

**Usage**

```
discriminator(  
    ch_in,  
    n_ftrs = 64,  
    n_layers = 3,  
    norm_layer = NULL,  
    sigmoid = FALSE  
)
```

**Arguments**

ch_in	input
n_ftrs	number of filters
n_layers	number of layers
norm_layer	normalization layer
sigmoid	apply sigmoid function or not

---

div	<i>Div</i>
-----	------------

---

**Description**

Div

**Usage**

```
## S3 method for class 'torch.Tensor'  
a / b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

DownmixMono

*Downmix Mono*


---

**Description**

Transform multichannel audios into single channel

**Usage**

```
DownmixMono(enc = NULL, dec = NULL, split_idx = NULL, order = NULL)
```

**Arguments**

enc	encoder
dec	decoder
split_idx	split by index
order	order, by default is NULL

**Value**

None

---

dropout\_mask

*Dropout\_mask*


---

**Description**

Return a dropout mask of the same type as 'x', size 'sz', with probability 'p' to cancel an element.

**Usage**

```
dropout_mask(x, sz, p)
```

**Arguments**

x	x
sz	sz
p	p

**Value**

None

---

 dummy\_eval

*Dummy\_eval*


---

**Description**

Evaluate 'm' on a dummy input of a certain 'size'

**Usage**

```
dummy_eval(m, size = list(64, 64))
```

**Arguments**

m	m parameter
size	size

**Value**

None

---

DynamicUnet

*DynamicUnet*


---

**Description**

Create a U-Net from a given architecture.

**Usage**

```
DynamicUnet(
  encoder,
  n_classes,
  img_size,
  blur = FALSE,
  blur_final = TRUE,
  self_attention = FALSE,
  y_range = NULL,
  last_cross = TRUE,
  bottle = FALSE,
  act_cls = nn()$ReLU,
  init = nn()$init$kaiming_normal_,
  norm_type = NULL
)
```

**Arguments**

encoder	encoder
n_classes	number of classes
img_size	image size
blur	blur is used to avoid checkerboard artifacts at each layer.
blur_final	blur final is specific to the last layer.
self_attention	self_attention determines if we use a self attention layer at the third block before the end.
y_range	If y_range is passed, the last activations go through a sigmoid rescaled to that range.
last_cross	last cross
bottle	bottle
act_cls	activation
init	initializer
norm_type	normalization type

**Value**

None

---

 EarlyStoppingCallback *EarlyStoppingCallback*


---

**Description**

EarlyStoppingCallback

**Usage**

EarlyStoppingCallback(...)

**Arguments**

... parameters to pass

**Value**

None

---

efficientdet\_infer\_dl *Efficientdet infer dataloader*

---

### Description

A ‘DataLoader’ with a custom ‘collate\_fn’ that batches items as required for inferring the model.

### Usage

```
efficientdet_infer_dl(dataset, batch_tfms = NULL, ...)
```

### Arguments

dataset	Possibly a ‘Dataset’ object, but more generally, any ‘Sequence’ that returns records.
batch_tfms	Transforms to be applied at the batch level. <b>**dataloader_kwargs</b> : Keyword arguments that will be internally passed to a Pytorch ‘DataLoader’. The parameter ‘collate_fn’ is already defined internally and cannot be passed here.
...	additional arguments

### Value

None

---

efficientdet\_learner *MaskRCNN learner*

---

### Description

Fastai ‘Learner’ adapted for MaskRCNN.

### Usage

```
efficientdet_learner(dls, model, cbs = NULL, ...)
```

### Arguments

dls	‘Sequence’ of ‘DataLoaders’ passed to the ‘Learner’. The first one will be used for training and the second for validation.
model	The model to train.
cbs	Optional ‘Sequence’ of callbacks.
...	learner_kwargs: Keyword arguments that will be internally passed to ‘Learner’.

### Value

model

---

efficientdet\_model     *Efficientdet model*

---

**Description**

Creates the efficientdet model specified by ‘model\_name’.

**Usage**

```
efficientdet_model(model_name, num_classes, img_size, pretrained = TRUE)
```

**Arguments**

model_name	Specifies the model to create. For pretrained models, check [this](https://github.com/rwightman/efficientdet-pytorch#models) table.
num_classes	Number of classes of your dataset (including background).
img_size	Image size that will be fed to the model. Must be squared and divisible by 128.
pretrained	If TRUE, use a pretrained backbone (on COCO).

**Value**

model

---

efficientdet\_predict\_dl  
*Efficientdet predict dataloader*

---

**Description**

Efficientdet predict dataloader

**Usage**

```
efficientdet_predict_dl(model, infer_dl, show_pbar = TRUE)
```

**Arguments**

model	model
infer_dl	infer_dl
show_pbar	show_pbar

**Value**

None



---

efficientdet\_train\_dl *Efficientdet train dataloader*

---

**Description**

A ‘DataLoader’ with a custom ‘collate\_fn’ that batches items as required for training the model.

**Usage**

```
efficientdet_train_dl(dataset, batch_tfms = NULL, ...)
```

**Arguments**

dataset	Possibly a ‘Dataset’ object, but more generally, any ‘Sequence’ that returns records.
batch_tfms	Transforms to be applied at the batch level.
...	dataloader_kwargs: Keyword arguments that will be internally passed to a PyTorch ‘DataLoader’. The parameter ‘collate_fn’ is already defined internally and cannot be passed here.

**Value**

None

---

efficientdet\_valid\_dl *Efficientdet valid dataloader*

---

**Description**

A ‘DataLoader’ with a custom ‘collate\_fn’ that batches items as required for training the model.

**Usage**

```
efficientdet_valid_dl(dataset, batch_tfms = NULL, ...)
```

**Arguments**

dataset	Possibly a ‘Dataset’ object, but more generally, any ‘Sequence’ that returns records.
batch_tfms	Transforms to be applied at the batch level.
...	dataloader_kwargs: Keyword arguments that will be internally passed to a PyTorch ‘DataLoader’. The parameter ‘collate_fn’ is already defined internally and cannot be passed here.

**Value**

None

---

Embedding

*Embedding*

---

**Description**

Embedding layer with truncated normal initialization

**Usage**

```
Embedding(ni, nf)
```

**Arguments**

ni	inputs
nf	outputs / number of features

**Value**

None

---

EmbeddingDropout

*EmbeddingDropout*

---

**Description**

Apply dropout with probability 'embed\_p' to an embedding layer 'emb'.

**Usage**

```
EmbeddingDropout(emb, embed_p)
```

**Arguments**

emb	emb
embed_p	embed_p

**Value**

None

---

emb_sz_rule	<i>Emb_sz_rule</i>
-------------	--------------------

---

**Description**

Rule of thumb to pick embedding size corresponding to 'n\_cat'

**Usage**

```
emb_sz_rule(n_cat)
```

**Arguments**

n_cat	n_cat
-------	-------

**Value**

None

---

error_rate	<i>Error rate</i>
------------	-------------------

---

**Description**

1 - 'accuracy'

**Usage**

```
error_rate(inp, targ, axis = -1)
```

**Arguments**

inp	The predictions of the model
targ	The corresponding labels
axis	Axis

**Value**

tensor

**Examples**

```

## Not run:

learn = cnn_learner(dls, resnet34(), metrics = error_rate)

## End(Not run)

```

---

exp	<i>Exp</i>
-----	------------

---

**Description**

Exp

**Usage**

```

## S3 method for class 'torch.Tensor'
exp(x)

```

**Arguments**

x            tensor

**Value**

tensor

---

exp.fastai.torch_core.TensorMask	<i>Exp</i>
----------------------------------	------------

---

**Description**

Exp

**Usage**

```

## S3 method for class 'fastai.torch_core.TensorMask'
exp(x)

```

**Arguments**

x                    tensor

**Value**

tensor

---

ExplainedVariance        *Explained Variance*

---

**Description**

Explained variance between predictions and targets

**Usage**

ExplainedVariance(sample\_weight = NULL)

**Arguments**

sample\_weight    sample\_weight

**Value**

None

---

expm1                    *Expm1*

---

**Description**

Expm1

**Usage**

```
## S3 method for class 'torch.Tensor'  
expm1(x)
```

**Arguments**

x                    tensor

**Value**

tensor

---

```
expm1.fastai.torch_core.TensorMask
```

*Expm1*

---

**Description**

Expm1

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'
expm1(x)
```

**Arguments**

x                    tensor

**Value**

tensor

---

```
export_generator        Export_generator
```

---

**Description**

Export\_generator

**Usage**

```
export_generator(
  learn,
  generator_name = "generator",
  path = ".",
  convert_to = "B"
)
```

**Arguments**

learn                learner/model  
generator\_name    generator name  
path                path (save dir)  
convert\_to        convert to

**Value**

None

---

exp_rmspe	<i>Exp_rmspe</i>
-----------	------------------

---

**Description**

Root mean square percentage error of the exponential of predictions and targets

**Usage**

```
exp_rmspe(preds, targs)
```

**Arguments**

preds	predicitons
targs	targets

**Value**

None

---

F1Score	<i>F1Score</i>
---------	----------------

---

**Description**

F1 score for single-label classification problems

**Usage**

```
F1Score(
  axis = -1,
  labels = NULL,
  pos_label = 1,
  average = "binary",
  sample_weight = NULL
)
```

**Arguments**

axis	axis
labels	labels
pos_label	pos_label
average	average
sample_weight	sample_weight

**Value**

None

---

`F1ScoreMulti`*F1ScoreMulti*

---

**Description**

F1 score for multi-label classification problems

**Usage**

```
F1ScoreMulti(  
    thresh = 0.5,  
    sigmoid = TRUE,  
    labels = NULL,  
    pos_label = 1,  
    average = "macro",  
    sample_weight = NULL  
)
```

**Arguments**

<code>thresh</code>	<code>thresh</code>
<code>sigmoid</code>	<code>sigmoid</code>
<code>labels</code>	<code>labels</code>
<code>pos_label</code>	<code>pos_label</code>
<code>average</code>	<code>average</code>
<code>sample_weight</code>	<code>sample_weight</code>

**Value**

None



---

fastai_version	<i>Fastai version</i>
----------------	-----------------------

---

**Description**

Fastai version

**Usage**

```
fastai_version()
```

**Value**

None

---

fastaudio	<i>Fastaudio module</i>
-----------	-------------------------

---

**Description**

Fastaudio module

**Usage**

```
fastaudio()
```

**Value**

None

---

faster_rcnn_infer_dl	<i>Faster RCNN infer dataloader</i>
----------------------	-------------------------------------

---

**Description**

A 'DataLoader' with a custom 'collate\_fn' that batches items as required for inferring the model.

**Usage**

```
faster_rcnn_infer_dl(dataset, batch_tfms = NULL, ...)
```

**Arguments**

dataset	Possibly a ‘Dataset‘ object, but more generally, any ‘Sequence‘ that returns records.
batch_tfms	Transforms to be applied at the batch level. <b>**dataloader_kwargs</b> : Keyword arguments that will be internally passed to a Pytorch ‘DataLoader‘. The parameter ‘collate_fn‘ is already defined internally and cannot be passed here.
...	additional arguments

**Value**

None

---

faster\_rcnn\_learner     *Faster RSNN learner*


---

**Description**

Fastai ‘Learner‘ adapted for Faster RCNN.

**Usage**

```
faster_rcnn_learner(dls, model, cbs = NULL, ...)
```

**Arguments**

dls	‘Sequence‘ of ‘DataLoaders‘ passed to the ‘Learner‘. The first one will be used for training and the second for validation.
model	The model to train.
cbs	Optional ‘Sequence‘ of callbacks.
...	<b>learner_kwargs</b> : Keyword arguments that will be internally passed to ‘Learner‘.

**Value**

model

---

faster\_rcnn\_model      *Faster RSNN model*

---

### Description

FasterRCNN model implemented by torchvision.

### Usage

```
faster_rcnn_model(
    num_classes,
    backbone = NULL,
    remove_internal_transforms = TRUE,
    pretrained = TRUE
)
```

### Arguments

num\_classes      Number of classes.

backbone      Backbone model to use. Defaults to a resnet50\_fpn model.

remove\_internal\_transforms

The torchvision model internally applies transforms like resizing and normalization, but we already do this at the 'Dataset' level, so it's safe to remove those internal transforms.

pretrained      Argument passed to 'fastercnn\_resnet50\_fpn' if 'backbone is NULL'. By default it is set to TRUE: this is generally used when training a new model (transfer learning). 'pretrained = FALSE' is used during inference (prediction) for cases where the users have their own pretrained weights. **\*\*faster\_rcnn\_kwargs**: Keyword arguments that internally are going to be passed to 'torchvision.models.detection.faster\_rcnn.FastRC

### Value

model

---

faster\_rcnn\_predict\_dl      *Faster RCNN predict dataloader*

---

### Description

Faster RCNN predict dataloader

### Usage

```
faster_rcnn_predict_dl(model, infer_dl, show_pbar = TRUE)
```

**Arguments**

model	model
infer_dl	infer_dl
show_pbar	show_pbar

**Value**

None

---

faster\_rcnn\_train\_dl *Faster RSNN train dataloader*

---

**Description**

A ‘DataLoader’ with a custom ‘collate\_fn’ that batches items as required for training the model.

**Usage**

```
faster_rcnn_train_dl(dataset, batch_tfms = NULL, ...)
```

**Arguments**

dataset	Possibly a ‘Dataset’ object, but more generally, any ‘Sequence’ that returns records.
batch_tfms	Transforms to be applied at the batch level.
...	dataloader_kwargs: Keyword arguments that will be internally passed to a PyTorch ‘DataLoader’. The parameter ‘collate_fn’ is already defined internally and cannot be passed here.

**Value**

None

---

faster\_rcnn\_valid\_dl *Faster RSNN valid dataloader*

---

### Description

A ‘DataLoader’ with a custom ‘collate\_fn’ that batches items as required for training the model.

### Usage

```
faster_rcnn_valid_dl(dataset, batch_tfms = NULL, ...)
```

### Arguments

dataset	Possibly a ‘Dataset’ object, but more generally, any ‘Sequence’ that returns records.
batch_tfms	Transforms to be applied at the batch level.
...	dataloader_kwargs: Keyword arguments that will be internally passed to a PyTorch ‘DataLoader’. The parameter ‘collate_fn’ is already defined internally and cannot be passed here.

### Value

None

---

fastinf *Wandb module*

---

### Description

Wandb module

### Usage

```
fastinf()
```

### Value

None

---

fa_collate	<i>Fa_collate</i>
------------	-------------------

---

**Description**

Fa\_collate

**Usage**

fa\_collate(t)

**Arguments**

t            text

**Value**

None

---

fa_convert	<i>Da_convert</i>
------------	-------------------

---

**Description**

Da\_convert

**Usage**

fa\_convert(t)

**Arguments**

t            text

**Value**

None

---

FBeta

*FBeta*

---

### Description

FBeta score with ‘beta’ for single-label classification problems

### Usage

```
FBeta(  
  beta,  
  axis = -1,  
  labels = NULL,  
  pos_label = 1,  
  average = "binary",  
  sample_weight = NULL  
)
```

### Arguments

beta	beta
axis	axis
labels	labels
pos_label	pos_label
average	average
sample_weight	sample_weight

### Value

None

---

FBetaMulti

*FBetaMulti*

---

### Description

FBeta score with ‘beta’ for multi-label classification problems

**Usage**

```
FBetaMulti(
  beta,
  thresh = 0.5,
  sigmoid = TRUE,
  labels = NULL,
  pos_label = 1,
  average = "macro",
  sample_weight = NULL
)
```

**Arguments**

beta	beta
thresh	thresh
sigmoid	sigmoid
labels	labels
pos_label	pos_label
average	average
sample_weight	sample_weight

**Value**

None

---

FetchPredsCallback	<i>FetchPredsCallback</i>
--------------------	---------------------------

---

**Description**

A callback to fetch predictions during the training loop

**Usage**

```
FetchPredsCallback(
  ds_idx = 1,
  dl = NULL,
  with_input = FALSE,
  with_decoded = FALSE,
  cbs = NULL,
  reorder = TRUE
)
```



**Arguments**

ds_idx	dataset index
dl	DL application
with_input	with input or not
with_decoded	with decoded or not
cbs	callbacks
reorder	reorder or not

**Value**

None

---

FileSplitter	<i>File Splitter</i>
--------------	----------------------

---

**Description**

Split 'items' by providing file 'fname' (contains names of valid items separated by newline).

**Usage**

```
FileSplitter(fname)
```

**Arguments**

fname	file name
-------	-----------

**Value**

None

---

FillMissing	<i>Fill Missing</i>
-------------	---------------------

---

**Description**

Fill the missing values in continuous columns.

**Usage**

```
FillMissing(
  cat_names,
  cont_names,
  fill_strategy = FillStrategy_MEDIAN(),
  add_col = TRUE,
  fill_val = 0
)
```

**Arguments**

cat_names	The names of the categorical variables
cont_names	The names of the continuous variables
fill_strategy	The strategy of filling
add_col	add_col
fill_val	fill_val

**Value**

None

**Examples**

```
## Not run:  
  
procs = list(FillMissing(),Categorify(),Normalize())  
  
## End(Not run)
```

---

FillStrategy\_COMMON    *COMMON*

---

**Description**

An enumeration.

**Usage**

```
FillStrategy_COMMON()
```

**Value**

None

---

FillStrategy\_CONSTANT    *CONSTANT*

---

**Description**

An enumeration.

**Usage**

FillStrategy\_CONSTANT()

**Value**

None

---

FillStrategy\_MEDIAN    *MEDIAN*

---

**Description**

An enumeration.

**Usage**

FillStrategy\_MEDIAN()

**Value**

None

---

find\_coeffs    *Find\_coeffs*

---

**Description**

Find coefficients for warp tfm from 'p1' to 'p2'

**Usage**

find\_coeffs(p1, p2)

**Arguments**

p1	coefficient p1
p2	coefficient p2

**Value**

None

---

 fine\_tune
 

---

*Fine\_tune***Description**

Fine tune with 'freeze' for 'freeze\_epochs' then with 'unfreeze' from 'epochs' using discriminative LR

**Usage**

```

fine_tune(
    object,
    epochs,
    base_lr = 0.002,
    freeze_epochs = 1,
    lr_mult = 100,
    pct_start = 0.3,
    div = 5,
    ...
)

```

**Arguments**

object	learner/model
epochs	epoch number
base_lr	base learning rate
freeze_epochs	freeze epochs number
lr_mult	learning rate multiply
pct_start	start percentage
div	divide
...	additional arguments

**Value**

None

---

```
fit.fastai.learner.Learner
```

*Fit*

---

**Description**

Fit the model on this learner with 'lr' learning rate, 'wd' weight decay for 'epochs' with 'callbacks' as cbs argument.

**Usage**

```
## S3 method for class 'fastai.learner.Learner'  
fit(object, ...)
```

**Arguments**

object	a learner object
...	parameters to pass

**Value**

train history

---

```
fit.fastai.tabular.learner.TabularLearner
```

*Fit*

---

**Description**

Fit the model on this learner with 'lr' learning rate, 'wd' weight decay for 'epochs' with 'callbacks'.

**Usage**

```
## S3 method for class 'fastai.tabular.learner.TabularLearner'  
fit(object, ...)
```

**Arguments**

object	model
...	additional arguments

**Value**

data frame

---

```
fit.fastai.vision.gan.GANLearner
    Fit
```

---

### Description

Fit the model on this learner with 'lr' learning rate, 'wd' weight decay for 'epochs' with 'callbacks'.

### Usage

```
## S3 method for class 'fastai.vision.gan.GANLearner'
fit(object, ...)
```

### Arguments

object	model
...	additional parameters to pass

### Value

train history

### Examples

```
## Not run:

learn %>% fit(1, 2e-4, wd = 0)

## End(Not run)
```

---

```
fit_flat_cos    Fit_flat_cos
```

---

### Description

Fit\_flat\_cos

**Usage**

```
fit_flat_cos(
    object,
    n_epoch,
    lr = NULL,
    div_final = 1e+05,
    pct_start = 0.75,
    wd = NULL,
    cbs = NULL,
    reset_opt = FALSE
)
```

**Arguments**

object	learner/model
n_epoch	number of epochs
lr	learning rate
div_final	divide final value
pct_start	start percentage
wd	weight decay
cbs	callbacks
reset_opt	reset optimizer

**Value**

None

---

fit_flat_lin	<i>Fit_flat_lin</i>
--------------	---------------------

---

**Description**

Fit 'self.model' for 'n\_epoch' at flat 'start\_lr' before 'curve\_type' annealing to 'end\_lr' with weight decay of 'wd' and callbacks 'cbs'.

**Usage**

```
fit_flat_lin(
    object,
    n_epochs = 100,
    n_epochs_decay = 100,
    start_lr = NULL,
    end_lr = 0,
    curve_type = "linear",
    wd = NULL,
```

```

    cbs = NULL,
    reset_opt = FALSE
)

```

### Arguments

object	model / learner
n_epochs	number of epochs
n_epochs_decay	number of epochs with decay
start_lr	Desired starting learning rate, used for beginning pct of training.
end_lr	Desired end learning rate, training will conclude at this learning rate.
curve_type	Curve type for learning rate annealing. Options are 'linear', 'cosine', and 'exponential'.
wd	weight decay
cbs	callbacks
reset_opt	reset optimizer

### Value

None

---

fit_one_cycle	<i>Fit one cycle</i>
---------------	----------------------

---

### Description

Fit one cycle

### Usage

```
fit_one_cycle(object, ...)
```

### Arguments

object	model
...	parameters to pass, e.g. lr, n_epoch, wd, and etc.

### Value

None



---

fit_sgdr	<i>Fit_sgdr</i>
----------	-----------------

---

**Description**

Fit\_sgdr

**Usage**

```
fit_sgdr(  
  object,  
  n_cycles,  
  cycle_len,  
  lr_max = NULL,  
  cycle_mult = 2,  
  cbs = NULL,  
  reset_opt = FALSE,  
  wd = NULL  
)
```

**Arguments**

object	learner/model
n_cycles	number of cycles
cycle_len	length of cycle
lr_max	maximum learning rate
cycle_mult	cycle mult
cbs	callbacks
reset_opt	reset optimizer
wd	weight decay

**Value**

None

---

FixedGANSwitcher	<i>Fixed GAN Switcher</i>
------------------	---------------------------

---

**Description**

Switcher to do 'n\_crit' iterations of the critic then 'n\_gen' iterations of the generator.

**Usage**

```
FixedGANSwitcher(n_crit = 1, n_gen = 1)
```

**Arguments**

n_crit	number of discriminator
n_gen	number of generator

**Value**

None

---

fix_fit	<i>Fix fit</i>
---------	----------------

---

**Description**

Fix fit

**Usage**

```
fix_fit(disable_graph = FALSE)
```

**Arguments**

disable_graph	to remove dynamic plot, by default is FALSE
---------------	---

**Value**

None

---

`fix_html`*Fix\_html*

---

**Description**

Various messy things we've seen in documents

**Usage**

```
fix_html(x)
```

**Arguments**

`x`                    `text`

**Value**

string

---

`Flatten`*Flatten*

---

**Description**

Flatten 'x' to a single dimension, e.g. at end of a model. 'full' for rank-1 tensor

**Usage**

```
Flatten(full = FALSE)
```

**Arguments**

`full`                    `bool, full or not`

---

flatten_check	<i>Flatten check</i>
---------------	----------------------

---

**Description**

Check that 'out' and 'targ' have the same number of elements and flatten them.

**Usage**

```
flatten_check(inp, targ)
```

**Arguments**

inp	predictions
targ	targets

**Value**

tensor

---

flatten_model	<i>Flatten_model</i>
---------------	----------------------

---

**Description**

Return the list of all submodules and parameters of 'm'

**Usage**

```
flatten_model(m)
```

**Arguments**

m	parameters
---	------------

**Value**

None

---

 Flip
*Flip***Description**

Randomly flip a batch of images with a probability ‘p’

**Usage**

```
Flip(
  p = 0.5,
  draw = NULL,
  size = NULL,
  mode = "bilinear",
  pad_mode = "reflection",
  align_corners = TRUE,
  batch = FALSE
)
```

**Arguments**

p	probability
draw	draw
size	size of image
mode	mode
pad_mode	reflection, zeros, border as string parameter
align_corners	align corners or not
batch	batch or not

**Value**

None

---

FlipItem

*FlipItem***Description**

Randomly flip with probability ‘p’

**Usage**

```
FlipItem(p = 0.5)
```

**Arguments**

p                    probability

**Value**

None

---

flip_mat	<i>Flip_mat</i>
----------	-----------------

---

**Description**

Return a random flip matrix

**Usage**

```
flip_mat(x, p = 0.5, draw = NULL, batch = FALSE)
```

**Arguments**

x                    tensor  
 p                    probability  
 draw                draw  
 batch                batch

**Value**

None

---

float	<i>Tensor to float</i>
-------	------------------------

---

**Description**

Tensor to float

**Usage**

```
float(tensor)
```

**Arguments**

tensor              tensor

**Value**

tensor

---

floor.fastai.torch\_core.TensorMask  
*Floor*

---

**Description**

Floor

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
floor(x)
```

**Arguments**

x                    tensor

**Value**

tensor

---

floor\_                    *Floor*

---

**Description**

Floor

**Usage**

```
## S3 method for class 'torch.Tensor'  
floor(x)
```

**Arguments**

x                    tensor

**Value**

tensor

---

floor\_div

*Floor divide*

---

**Description**

Floor divide

**Usage**

```
## S3 method for class 'torch.Tensor'  
x %% y
```

**Arguments**

x	tensor
y	tensor

**Value**

tensor

---

floor\_mod

*Floor mod*

---

**Description**

Floor mod

**Usage**

```
## S3 method for class 'torch.Tensor'  
x %% y
```

**Arguments**

x	tensor
y	tensor

**Value**

tensor



---

fmodule	<i>Module</i>
---------	---------------

---

**Description**

Module

**Usage**

fmodule(...)

**Arguments**

... parameters to pass

**Details**

Decorator to create an nn()\$Module using f as forward method

**Value**

None

---

FolderDataset	<i>FolderDataset</i>
---------------	----------------------

---

**Description**

A PyTorch Dataset class that can be created from a folder ‘path’ of images, for the sole purpose of inference. Optional ‘transforms’

**Usage**

FolderDataset(path, transforms = NULL)

**Arguments**

path	path to dir
transforms	transformations

**Details**

can be provided. Attributes: ‘self.files’: A list of the filenames in the folder. ‘self.totensor’: ‘torchvision.transforms.ToTensor’ transform. ‘self.transform’: The transforms passed in as ‘transforms’ to the constructor.

**Value**

None

---

force_plot	<i>Force_plot</i>
------------	-------------------

---

**Description**

Visualizes the SHAP values with an added force layout. Accepts a `class_id` which is used to indicate the class of interest for a classification model.

**Usage**

```
force_plot(object, class_id = 0, ...)
```

**Arguments**

<code>object</code>	ShapInterpretation object
<code>class_id</code>	Accepts a <code>class_id</code> which is used to indicate the class of interest for a classification model. It can either be an int or str representation for a class of choice.
<code>...</code>	additional arguments

**Value**

None

---

foreground_acc	<i>Foreground accuracy</i>
----------------	----------------------------

---

**Description**

Computes non-background accuracy for multiclass segmentation

**Usage**

```
foreground_acc(inp, targ, bkg_idx = 0, axis = 1)
```

**Arguments**

<code>inp</code>	predictions
<code>targ</code>	targets
<code>bkg_idx</code>	bkg_idx
<code>axis</code>	axis

**Value**

None

---

ForgetMultGPU	<i>ForgetMultGPU</i>
---------------	----------------------

---

**Description**

Wrapper around the CUDA kernels for the ForgetMult gate.

**Usage**

```
ForgetMultGPU(...)
```

**Arguments**

... parameters to pass

**Value**

None

---

forget_mult_CPU	<i>Forget_mult_CPU</i>
-----------------	------------------------

---

**Description**

ForgetMult gate applied to 'x' and 'f' on the CPU.

**Usage**

```
forget_mult_CPU(x, f, first_h = NULL, batch_first = TRUE, backward = FALSE)
```

**Arguments**

x	x
f	f
first_h	first_h
batch_first	batch_first
backward	backward

**Value**

None

---

freeze	<i>Freeze a model</i>
--------	-----------------------

---

**Description**

Freeze a model

**Usage**

```
freeze(object, ...)
```

**Arguments**

object	A model
...	Additional parameters

**Value**

None

**Examples**

```
## Not run:  
learnR %>% freeze()  
  
## End(Not run)
```

---

FuncSplitter	<i>FuncSplitter</i>
--------------	---------------------

---

**Description**

Split 'items' by result of 'func' ('TRUE' for validation, 'FALSE' for training set).

**Usage**

```
FuncSplitter(func)
```

**Arguments**

func	function
------	----------

**Value**

None

---

`fView``View`

---

**Description**

Reshape x to size

**Usage**

```
fView(...)
```

**Arguments**

`...` parameters to pass

**Value**

None

---

`GANDiscriminativeLR``GAN Discriminative LR`

---

**Description**

‘Callback’ that handles multiplying the learning rate by ‘mult\_lr’ for the critic.

**Usage**

```
GANDiscriminativeLR(mult_lr = 5)
```

**Arguments**

`mult_lr` mult learning rate

---

GANLearner\_from\_learners

*GAN Learner from learners*


---

### Description

Create a GAN from ‘learn\_gen‘ and ‘learn\_crit‘.

### Usage

```
GANLearner_from_learners(
    gen_learn,
    crit_learn,
    switcher = NULL,
    weights_gen = NULL,
    gen_first = FALSE,
    switch_eval = TRUE,
    show_img = TRUE,
    clip = NULL,
    cbs = NULL,
    metrics = NULL,
    loss_func = NULL,
    opt_func = Adam(),
    lr = 0.001,
    splitter = trainable_params(),
    path = NULL,
    model_dir = "models",
    wd = NULL,
    wd_bn_bias = FALSE,
    train_bn = TRUE,
    moms = list(0.95, 0.85, 0.95)
)
```

### Arguments

gen_learn	generator learner
crit_learn	discriminator learner
switcher	switcher
weights_gen	weights generator
gen_first	generator first
switch_eval	switch evaluation
show_img	show image or not
clip	clip value
cbs	Cbs is one or a list of Callbacks to pass to the Learner.

metrics	It is an optional list of metrics, that can be either functions or Metrics.
loss_func	loss function
opt_func	The function used to create the optimizer
lr	learning rate
splitter	It is a function that takes self.model and returns a list of parameter groups (or just one parameter group if there are no different parameter groups).
path	The folder where to work
model_dir	Path and model_dir are used to save and/or load models.
wd	It is the default weight decay used when training the model.
wd_bn_bias	It controls if weight decay is applied to BatchNorm layers and bias.
train_bn	It controls if BatchNorm layers are trained even when they are supposed to be frozen according to the splitter.
moms	The default momentums used in Learner\$fit_one_cycle.

**Value**

None

---

GANLearner\_wgan      *Wgan*


---

**Description**

Create a WGAN from 'data', 'generator' and 'critic'.

**Usage**

```
GANLearner_wgan(
  dls,
  generator,
  critic,
  switcher = NULL,
  clip = 0.01,
  switch_eval = FALSE,
  gen_first = FALSE,
  show_img = TRUE,
  cbs = NULL,
  metrics = NULL,
  opt_func = Adam(),
  lr = 0.001,
  splitter = trainable_params,
  path = NULL,
  model_dir = "models",
  wd = NULL,
```

```

    wd_bn_bias = FALSE,
    train_bn = TRUE,
    moms = list(0.95, 0.85, 0.95)
)

```

### Arguments

dls	dataloader
generator	generator
critic	critic
switcher	switcher
clip	clip value
switch_eval	switch evaluation
gen_first	generator first
show_img	show image or not
cbs	callbacks
metrics	metrics
opt_func	optimization function
lr	learning rate
splitter	splitter
path	path
model_dir	model directory
wd	weight decay
wd_bn_bias	weight decay bn bias
train_bn	It controls if BatchNorm layers are trained even when they are supposed to be frozen according to the splitter.
moms	momentums

### Value

None

### Examples

```

## Not run:

learn = GANLearner_wgan(dls, generator, critic, opt_func = partial(Adam(), mom=0.))

## End(Not run)

```



---

`GANLoss`*GAN Loss*

---

**Description**

Wrapper around 'crit\_loss\_func' and 'gen\_loss\_func'

**Usage**

```
GANLoss(gen_loss_func, crit_loss_func, gan_model)
```

**Arguments**

<code>gen_loss_func</code>	generator loss function
<code>crit_loss_func</code>	discriminator loss function
<code>gan_model</code>	GAN model

**Value**

None

---

`GANModule`*GAN Module*

---

**Description**

Wrapper around a 'generator' and a 'critic' to create a GAN.

**Usage**

```
GANModule(generator = NULL, critic = NULL, gen_mode = FALSE)
```

**Arguments**

<code>generator</code>	generator
<code>critic</code>	critic
<code>gen_mode</code>	generator mode or not

**Value**

None

---

GANTrainer

*GAN Trainer*


---

**Description**

Handles GAN Training.

**Usage**

```
GANTrainer(
    switch_eval = FALSE,
    clip = NULL,
    beta = 0.98,
    gen_first = FALSE,
    show_img = TRUE
)
```

**Arguments**

switch_eval	switch evaluation
clip	clip value
beta	beta parameter
gen_first	generator first
show_img	show image or not

**Value**

None

---

gan\_critic

*Gan critic*


---

**Description**

Critic to train a 'GAN'.

**Usage**

```
gan_critic(n_channels = 3, nf = 128, n_blocks = 3, p = 0.15)
```

**Arguments**

n_channels	number of channels
nf	number of features
n_blocks	number of blocks
p	probability

**Value**

GAN object

---

gan\_loss\_from\_func     *GAN loss from function*

---

**Description**

Define loss functions for a GAN from 'loss\_gen' and 'loss\_crit'.

**Usage**

```
gan_loss_from_func(loss_gen, loss_crit, weights_gen = NULL)
```

**Arguments**

loss_gen	generator loss
loss_crit	discriminator loss
weights_gen	weight generator

**Value**

None

---

GatherPredsCallback     *GatherPredsCallback*

---

**Description**

'Callback' that saves the predictions and targets, optionally 'with\_loss'

**Usage**

```
GatherPredsCallback(  
  with_input = FALSE,  
  with_loss = FALSE,  
  save_preds = NULL,  
  save_targs = NULL,  
  concat_dim = 0  
)
```

**Arguments**

with_input	include inputs or not
with_loss	include loss or not
save_preds	save predictions
save_targs	save targets/actuals
concat_dim	concatenate dimensions

**Value**

None

---

gauss_blur2d	<i>Gauss_blur2d</i>
--------------	---------------------

---

**Description**

Apply gaussian\_blur2d kornia filter

**Usage**

gauss\_blur2d(x, s)

**Arguments**

x	image
s	effect

**Value**

None

---

generate_noise	<i>Generate noise</i>
----------------	-----------------------

---

**Description**

Generate noise

**Usage**

generate\_noise(fn, size = 100)

**Arguments**

fn	path
size	the size

**Value**

None

**Examples**

```
## Not run:  
generate_noise()  
  
## End(Not run)
```

---

<code>get_annotations</code>	<i>Get_annotations</i>
------------------------------	------------------------

---

**Description**

Open a COCO style json in 'fname' and returns the lists of filenames (with maybe 'prefix') and labelled bboxes.

**Usage**

```
get_annotations(fname, prefix = NULL)
```

**Arguments**

fname	folder name
prefix	prefix

**Value**

None

---

get_audio_files	<i>Get_audio_files</i>
-----------------	------------------------

---

**Description**

Get audio files in 'path' recursively, only in 'folders', if specified.

**Usage**

```
get_audio_files(path, recurse = TRUE, folders = NULL)
```

**Arguments**

path	path
recurse	recursive or not
folders	vector, folders

**Value**

None

---

get_bias	<i>Get_bias</i>
----------	-----------------

---

**Description**

Bias for item or user (based on 'is\_item') for all in 'arr'

**Usage**

```
get_bias(object, arr, is_item = TRUE, convert = TRUE)
```

**Arguments**

object	extract bias
arr	R data frame
is_item	logical, is item
convert	to R matrix

**Value**

tensor

**Examples**

```
## Not run:  
  
movie_bias = learn %>% get_bias(top_movies, is_item = TRUE)  
  
## End(Not run)
```

---

get\_c

*Get\_c*

---

**Description**

Get\_c

**Usage**

```
get_c(dls)
```

**Arguments**

dls                    dataloader object

**Value**

number of layers

**Examples**

```
## Not run:  
  
get_c(dls)  
  
## End(Not run)
```

---

get\_confusion\_matrix *Extract confusion matrix*

---

**Description**

Extract confusion matrix

**Usage**

```
get_confusion_matrix(object)
```

**Arguments**

object            model

**Value**

matrix

**Examples**

```
## Not run:  
  
model %>% get_confusion_matrix()  
  
## End(Not run)
```

---

get\_data\_loaders        *Get data loaders*

---

**Description**

Get data loaders

**Usage**

```
get_data_loaders(train_batch_size, val_batch_size)
```

**Arguments**

train\_batch\_size        train dataset batch size  
val\_batch\_size        validation dataset batch size



**Value**

None

---

get_dcm_matrix	<i>Get image matrix</i>
----------------	-------------------------

---

**Description**

Get image matrix

**Usage**

```
get_dcm_matrix(img, type = "raw", scan = "", size = 50, convert = TRUE)
```

**Arguments**

img	dicom file
type	img transformation
scan	apply uniform or gaussian blur effects
size	size of image
convert	to R matrix or keep tensor

**Value**

tensor

**Examples**

```
## Not run:  
  
img = dcmread('hemorrhage.dcm')  
img %>% get_dcm_matrix(type = 'raw')  
  
## End(Not run)
```

---

get_dicom_files	<i>get_dicom_files</i>
-----------------	------------------------

---

**Description**

Get dicom files in 'path' recursively, only in 'folders', if specified.

**Usage**

```
get_dicom_files(path, recurse = TRUE, folders = NULL)
```

**Arguments**

path	path to files
recurse	recursive or not
folders	folder names

**Value**

lsit of files

**Examples**

```
## Not run:  
  
items = get_dicom_files("siim_small/train/")  
  
## End(Not run)
```

---

get_dls	<i>Get dls</i>
---------	----------------

---

**Description**

Given image files from two domains ('pathA', 'pathB'), create 'DataLoaders' object.

**Usage**

```

get_dls(
    pathA,
    pathB,
    num_A = NULL,
    num_B = NULL,
    load_size = 512,
    crop_size = 256,
    bs = 4,
    num_workers = 2
)

```

**Arguments**

pathA	path A (from domain)
pathB	path B (to domain)
num_A	subset of A data
num_B	subset of B data
load_size	load size
crop_size	crop size
bs	batch size
num_workers	number of workers

**Details**

Loading and randomly cropped sizes of 'load\_size' and 'crop\_size' are set to defaults of 512 and 256. Batch size is specified by 'bs' (default=4).

**Value**

None

---

get\_emb\_sz

*Get\_emb\_sz*

---

**Description**

Get default embedding size from 'TabularPreprocessor' 'proc' or the ones in 'sz\_dict'

**Usage**

```
get_emb_sz(to, sz_dict = NULL)
```

**Arguments**

to	to
sz_dict	dictionary size

**Value**

None

---

get_files	<i>Get_files</i>
-----------	------------------

---

**Description**

Get all the files in 'path' with optional 'extensions', optionally with 'recurse', only in 'folders', if specified.

**Usage**

```
get_files(
    path,
    extensions = NULL,
    recurse = TRUE,
    folders = NULL,
    followlinks = TRUE
)
```

**Arguments**

path	path
extensions	extensions
recurse	recurse
folders	folders
followlinks	followlinks

**Value**

list

---

get_grid	<i>Get_grid</i>
----------	-----------------

---

**Description**

Return a grid of 'n' axes, 'rows' by 'cols'

**Usage**

```
get_grid(  
    n,  
    nrows = NULL,  
    ncols = NULL,  
    add_vert = 0,  
    figsize = NULL,  
    double = FALSE,  
    title = NULL,  
    return_fig = FALSE,  
    imsize = 3  
)
```

**Arguments**

n	n
nrows	number of rows
ncols	number of columns
add_vert	add vertical
figsize	figure size
double	double
title	title
return_fig	return figure or not
imsize	image size

**Value**

None

---

<code>get_hf_objects</code>	<i>Get_hf_objects</i>
-----------------------------	-----------------------

---

**Description**

Returns the architecture (str), config (obj), tokenizer (obj), and model (obj) given at minimum a

**Usage**

```
get_hf_objects(...)
```

**Arguments**

... parameters to pass

**Details**

'pre-trained model name or path'. Specify a 'task' to ensure the right "AutoModelFor<task>" is used to create the model. Optionally, you can pass a config (obj), tokenizer (class), and/or model (class) (along with any related kwargs for each) to get as specific as you want w/r/t what huggingface objects are returned.

**Value**

None

---

<code>get_image_files</code>	<i>Get image files</i>
------------------------------	------------------------

---

**Description**

Get image files in 'path' recursively, only in 'folders', if specified.

**Usage**

```
get_image_files(path, recurse = TRUE, folders = NULL)
```

**Arguments**

<code>path</code>	The folder where to work
<code>recurse</code>	recursive path
<code>folders</code>	folder names

**Value**

None

**Examples**

```
## Not run:  
  
URLs_PETS()  
  
path = 'oxford-iiit-pet'  
  
path_img = 'oxford-iiit-pet/images'  
fnames = get_image_files(path_img)  
  
## End(Not run)
```

---

`get_language_model`      *Get\_language\_model*

---

**Description**

Create a language model from ‘arch’ and its ‘config’.

**Usage**

```
get_language_model(arch, vocab_sz, config = NULL, drop_mult = 1)
```

**Arguments**

<code>arch</code>	<code>arch</code>
<code>vocab_sz</code>	<code>vocab_sz</code>
<code>config</code>	<code>config</code>
<code>drop_mult</code>	<code>drop_mult</code>

**Value**

model

---

get\_preds\_cyclegan      *Get\_preds\_cyclegan*

---

### Description

A prediction function that takes the Learner object ‘learn’ with the trained model, the ‘test\_path’ folder with the images to perform

### Usage

```
get_preds_cyclegan(
    learn,
    test_path,
    pred_path,
    bs = 4,
    num_workers = 4,
    suffix = "tif"
)
```

### Arguments

learn	learner/model
test_path	testdat path
pred_path	predict data path
bs	batch size
num_workers	number of workers
suffix	suffix

### Details

batch inference on, and the output folder ‘pred\_path’ where the predictions will be saved, with a batch size ‘bs’, ‘num\_workers’, and suffix of the prediction images ‘suffix’ (default=’png’).

---

get\_text\_classifier      *Get\_text\_classifier*

---

### Description

Create a text classifier from ‘arch’ and its ‘config’, maybe ‘pretrained’



**Usage**

```

get_text_classifier(
    arch,
    vocab_sz,
    n_class,
    seq_len = 72,
    config = NULL,
    drop_mult = 1,
    lin_ftrs = NULL,
    ps = NULL,
    pad_idx = 1,
    max_len = 1440,
    y_range = NULL
)

```

**Arguments**

arch	arch
vocab_sz	vocab_sz
n_class	n_class
seq_len	seq_len
config	config
drop_mult	drop_mult
lin_ftrs	lin_ftrs
ps	ps
pad_idx	pad_idx
max_len	max_len
y_range	y_range

**Value**

None

---

get_text_files	<i>Get_text_files</i>
----------------	-----------------------

---

**Description**

Get text files in 'path' recursively, only in 'folders', if specified.

**Usage**

```
get_text_files(path, recurse = TRUE, folders = NULL)
```

**Arguments**

path	path
recurse	recurse
folders	folders

**Value**

None

---

get_weights	<i>Get weights</i>
-------------	--------------------

---

**Description**

Weight for item or user (based on 'is\_item') for all in 'arr'

**Usage**

```
get_weights(object, arr, is_item = TRUE, convert = FALSE)
```

**Arguments**

object	extract weights
arr	R data frame
is_item	logical, is item
convert	to R matrix

**Value**

tensor

**Examples**

```
## Not run:  
  
movie_w = learn %>% get_weights(top_movies, is_item = TRUE, convert = TRUE)  
  
## End(Not run)
```

---

GradientAccumulation    *GradientAccumulation*

---

**Description**

Accumulate gradients before updating weights

**Usage**

```
GradientAccumulation(n_acc = 32)
```

**Arguments**

n\_acc            number of acc

**Value**

None

---

GrandparentSplitter    *GrandparentSplitter*

---

**Description**

Split 'items' from the grand parent folder names ('train\_name' and 'valid\_name').

**Usage**

```
GrandparentSplitter(train_name = "train", valid_name = "valid")
```

**Arguments**

train\_name        train folder name  
valid\_name        validation folder name

**Value**

None

grayscale

*Grayscale*

---

**Description**

Tensor to grayscale tensor. Uses the ITU-R 601-2 luma transform.

**Usage**

```
grayscale(x)
```

**Arguments**

x            tensor

**Value**

None

---

greater

*Greater*

---

**Description**

Greater

**Usage**

```
## S3 method for class 'torch.Tensor'  
a > b
```

**Arguments**

a            tensor

b            tensor

**Value**

tensor

---

greater_or_equal	<i>Greater or equal</i>
------------------	-------------------------

---

**Description**

Greater or equal

**Usage**

```
## S3 method for class 'torch.Tensor'
a >= b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

HammingLoss	<i>HammingLoss</i>
-------------	--------------------

---

**Description**

Hamming loss for single-label classification problems  
 Hamming loss for single-label classification problems

**Usage**

```
HammingLoss(axis = -1, sample_weight = NULL)
HammingLoss(axis = -1, sample_weight = NULL)
```

**Arguments**

axis	axis
sample_weight	sample_weight

**Value**

Loss object  
 None

---

HammingLossMulti	<i>HammingLossMulti</i>
------------------	-------------------------

---

**Description**

Hamming loss for multi-label classification problems

**Usage**

```
HammingLossMulti(
    thresh = 0.5,
    sigmoid = TRUE,
    labels = NULL,
    sample_weight = NULL
)
```

**Arguments**

thresh	threshold
sigmoid	sigmoid
labels	labels
sample_weight	sample_weight

**Value**

Loss object

---

has_params	<i>Has_params</i>
------------	-------------------

---

**Description**

Check if 'm' has at least one parameter

**Usage**

```
has_params(m)
```

**Arguments**

m	m parameter
---	-------------

**Value**

None

---

has_pool_type	<i>Has_pool_type</i>
---------------	----------------------

---

**Description**

Return 'TRUE' if 'm' is a pooling layer or has one in its children

**Usage**

has\_pool\_type(m)

**Arguments**

m	parameters
---	------------

**Value**

None

---

helper	<i>BLURR_MODEL_HELPER</i>
--------	---------------------------

---

**Description**

BLURR\_MODEL\_HELPER

**Usage**

helper()

**Value**

None

---

HF_ARCHITECTURES	<i>HF_ARCHITECTURES</i>
------------------	-------------------------

---

**Description**

An enumeration.

**Usage**

HF\_ARCHITECTURES()

**Value**

None

---

HF\_BaseInput

*HF\_BaseInput*

---

### Description

A HF\_BaseInput object is returned from the decodes method of HF\_BatchTransform as a mean to customize '@typedispatched' functions like DataLoaders.show\_batch and Learner.show\_results. It represents the "input\_ids" of a huggingface sequence as a tensor with a show method that requires a huggingface tokenizer for proper display.

### Usage

```
HF_BaseInput(...)
```

### Arguments

... parameters to pass

### Value

None

---

HF\_BaseModelCallback

*HF\_BaseModelCallback*

---

### Description

HF\_BaseModelCallback

### Usage

```
HF_BaseModelCallback(...)
```

### Arguments

... parameters to pass

### Value

None



---

HF\_BaseModelWrapper    *HF\_BaseModelWrapper*

---

### Description

Same as 'nn.Module', but no need for subclasses to call 'super().\_\_init\_\_'

### Usage

```
HF_BaseModelWrapper(  
    hf_model,  
    output_hidden_states = FALSE,  
    output_attentions = FALSE,  
    ...  
)
```

### Arguments

hf_model	model
output_hidden_states	output hidden states
output_attentions	output attentions
...	additional arguments to pass

### Value

None

---

HF\_BeforeBatchTransform  
*HF\_BeforeBatchTransform*

---

### Description

Handles everything you need to assemble a mini-batch of inputs and targets, as well as decode the dictionary produced as a byproduct of the tokenization process in the 'encodes' method.

**Usage**

```
HF_BeforeBatchTransform(
    hf_arch,
    hf_tokenizer,
    max_length = NULL,
    padding = TRUE,
    truncation = TRUE,
    is_split_into_words = FALSE,
    n_tok_inps = 1,
    ...
)
```

**Arguments**

hf_arch	architecture
hf_tokenizer	tokenizer
max_length	maximum length
padding	padding or not
truncation	truncation or not
is_split_into_words	to split into words
n_tok_inps	number tok inputs
...	additional arguments

**Value**

None

---

HF\_CausalLMBeforeBatchTransform

*HF\_CausalLMBeforeBatchTransform*

---

**Description**

Handles everything you need to assemble a mini-batch of inputs and targets, as well as decode the dictionary produced

**Usage**

```
HF_CausalLMBeforeBatchTransform(
    hf_arch,
    hf_tokenizer,
    max_length = NULL,
    padding = TRUE,
    truncation = TRUE,
```

```

    is_split_into_words = FALSE,
    n_tok_inps = 1,
    ignore_token_id = -100,
    ...
)

```

### Arguments

hf_arch	architecture
hf_tokenizer	tokenizer
max_length	maximum length
padding	padding or not
truncation	truncation or not
is_split_into_words	to split into words
n_tok_inps	number tok inputs
ignore_token_id	ignore token id
...	additional arguments

### Details

as a byproduct of the tokenization process in the ‘encodes’ method.

### Value

None

---

HF_load_dataset	<i>Load_dataset</i>
-----------------	---------------------

---

### Description

Load a dataset

### Usage

```

HF_load_dataset(
  path,
  name = NULL,
  data_dir = NULL,
  data_files = NULL,
  split = NULL,
  cache_dir = NULL,
  features = NULL,
  download_config = NULL,
)

```

```

    download_mode = NULL,
    ignore_verifications = FALSE,
    save_infos = FALSE,
    script_version = NULL,
    ...
)

```

### Arguments

path	path
name	name
data_dir	dataset dir
data_files	dataset files
split	split
cache_dir	cache directory
features	features
download_config	download configuration
download_mode	download mode
ignore_verifications	ignore verifications or not
save_infos	save information or not
script_version	script version
...	additional arguments

### Details

This method does the following under the hood: 1. Download and import in the library the dataset loading script from “path“ if it’s not already cached inside the library. Processing scripts are small python scripts that define the citation, info and format of the dataset, contain the URL to the original data files and the code to load examples from the original data files. You can find some of the scripts here: <https://github.com/huggingface/datasets/datasets> and easily upload yours to share them using the CLI “datasets-cli“. 2. Run the dataset loading script which will: \* Download the dataset file from the original URL (see the script) if it’s not already downloaded and cached. \* Process and cache the dataset in typed Arrow tables for caching. Arrow table are arbitrarily long, typed tables which can store nested objects and be mapped to numpy/pandas/python standard types. They can be directly access from drive, loaded in RAM or even streamed over the web. 3. Return a dataset build from the requested splits in “split“ (default: all).

### Value

data frame

---

HF\_QABatchTransform    *HF\_QABatchTransform*

---

### Description

Handles everything you need to assemble a mini-batch of inputs and targets, as well as decode the dictionary produced

### Usage

```
HF_QABatchTransform(  
    hf_arch,  
    hf_tokenizer,  
    max_length = NULL,  
    padding = TRUE,  
    truncation = TRUE,  
    is_split_into_words = FALSE,  
    n_tok_inps = 1,  
    hf_input_return_type = HF_QuestionAnswerInput(),  
    ...  
)
```

### Arguments

hf_arch	architecture
hf_tokenizer	tokenizer
max_length	maximum length
padding	padding
truncation	truncation
is_split_into_words	to split into words or not
n_tok_inps	number of tok inputs
hf_input_return_type	input return type
...	additional arguments

### Details

as a byproduct of the tokenization process in the ‘encodes’ method.

### Value

None

---

HF\_QABeforeBatchTransform

*HF\_QABeforeBatchTransform*


---

### Description

Handles everything you need to assemble a mini-batch of inputs and targets, as well as decode the dictionary produced

### Usage

```
HF_QABeforeBatchTransform(
    hf_arch,
    hf_tokenizer,
    max_length = NULL,
    padding = TRUE,
    truncation = TRUE,
    is_split_into_words = FALSE,
    n_tok_inps = 1,
    ...
)
```

### Arguments

hf_arch	architecture
hf_tokenizer	tokenizer
max_length	maximum length
padding	padding or not
truncation	truncation or not
is_split_into_words	into split into words or not
n_tok_inps	number of tok inputs
...	additional arguments

### Details

as a byproduct of the tokenization process in the ‘encodes’ method.

### Value

None

---

HF\_QstAndAnsModelCallback

*HF\_QstAndAnsModelCallback*

---

**Description**

HF\_QstAndAnsModelCallback

**Usage**

HF\_QstAndAnsModelCallback(...)

**Arguments**

... parameters to pass

**Value**

None

---

HF\_QuestionAnswerInput

*HF\_QuestionAnswerInput*

---

**Description**

HF\_QuestionAnswerInput

**Usage**

HF\_QuestionAnswerInput(...)

**Arguments**

... parameters to apss

**Value**

None

---

hf_splitter	<i>Hf_splitter</i>
-------------	--------------------

---

**Description**

Splits the huggingface model based on various model architecture conventions

**Usage**

```
hf_splitter(m)
```

**Arguments**

m	parameters
---	------------

**Value**

None

---

HF_SummarizationBeforeBatchTransform	<i>HF_SummarizationBeforeBatchTransform</i>
--------------------------------------	---

---

**Description**

Handles everything you need to assemble a mini-batch of inputs and targets, as well as decode the dictionary produced as a byproduct of the tokenization process in the 'encodes' method.

**Usage**

```
HF_SummarizationBeforeBatchTransform(
    hf_arch,
    hf_tokenizer,
    max_length = NULL,
    padding = TRUE,
    truncation = TRUE,
    is_split_into_words = FALSE,
    n_tok_inps = 2,
    ignore_token_id = -100,
    ...
)
```



**Arguments**

<code>hf_arch</code>	architecture
<code>hf_tokenizer</code>	tokenizer
<code>max_length</code>	maximum length
<code>padding</code>	padding or not
<code>truncation</code>	truncation or not
<code>is_split_into_words</code>	to split into words
<code>n_tok_inps</code>	number tok inputs
<code>ignore_token_id</code>	ignore token id
<code>...</code>	additional arguments

**Value**

None

---

`HF_SummarizationInput` *HF\_SummarizationInput*

---

**Description**

`HF_SummarizationInput`

**Usage**

`HF_SummarizationInput()`

**Value**

None

---

HF\_SummarizationModelCallback

*HF\_SummarizationModelCallback*


---

**Description**

Basic class handling tweaks of the training loop by changing a ‘Learner‘ in various events

**Usage**

```
HF_SummarizationModelCallback(
  rouge_metrics = c("rouge1", "rouge2", "rougeL"),
  ignore_token_id = -100,
  ...
)
```

**Arguments**

rouge_metrics	rouge metrics
ignore_token_id	integer, ignore token id
...	additional arguments

**Value**

None

---

HF\_TASKS\_ALL

*HF\_TASKS\_ALL*


---

**Description**

An enumeration.

**Usage**

```
HF_TASKS_ALL()
```

**Value**

None

---

HF_TASKS_AUTO	<i>HF_TASKS_AUTO</i>
---------------	----------------------

---

**Description**

An enumeration.

**Usage**

```
HF_TASKS_AUTO()
```

**Value**

None

---

HF_Text2TextAfterBatchTransform	<i>HF_Text2TextAfterBatchTransform</i>
---------------------------------	--

---

**Description**

Delegates ('\_\_call\_\_', 'decode', 'setup') to (`encodes`, `decodes`, `setups`) if 'split\_idx' matches

**Usage**

```
HF_Text2TextAfterBatchTransform(
    hf_tokenizer,
    input_return_type = HF_BaseInput()
)
```

**Arguments**

hf_tokenizer	tokenizer
input_return_type	input return type

**Value**

None

HF\_Text2TextBlock      *HF\_Text2TextBlock*

---

**Description**

A basic wrapper that links defaults transforms for the data block API

**Usage**

```
HF_Text2TextBlock(...)
```

**Arguments**

...                    parameters to pass

**Value**

None

---

HF\_TextBlock            *HF\_TextBlock*

---

**Description**

A basic wrapper that links defaults transforms for the data block API

**Usage**

```
HF_TextBlock(...)
```

**Arguments**

...                    arguments to pass

**Value**

None

---

HF\_TokenCategorize     *HF\_TokenCategorize*

---

**Description**

Reversible transform of a list of category string to 'vocab' id

**Usage**

```
HF_TokenCategorize(vocab = NULL, ignore_token = NULL, ignore_token_id = NULL)
```

**Arguments**

vocab	vocabulary
ignore_token	ignore token
ignore_token_id	ignore token id

**Value**

None

---

HF\_TokenCategoryBlock     *HF\_TokenCategoryBlock*

---

**Description**

'TransformBlock' for single-label categorical targets

**Usage**

```
HF_TokenCategoryBlock(  
  vocab = NULL,  
  ignore_token = NULL,  
  ignore_token_id = NULL  
)
```

**Arguments**

vocab	vocabulary
ignore_token	ignore token
ignore_token_id	ignore token id

**Value**

None

---

HF-TokenClassBeforeBatchTransform

*HF-TokenClassBeforeBatchTransform*


---

### Description

Handles everything you need to assemble a mini-batch of inputs and targets, as well as decode the dictionary produced

### Usage

```
HF-TokenClassBeforeBatchTransform(
    hf_arch,
    hf_tokenizer,
    ignore_token_id = -100,
    max_length = NULL,
    padding = TRUE,
    truncation = TRUE,
    is_split_into_words = TRUE,
    n_tok_inps = 1,
    ...
)
```

### Arguments

hf_arch	architecture
hf_tokenizer	tokenizer
ignore_token_id	ignore token id
max_length	maximum length
padding	padding or not
truncation	truncation or not
is_split_into_words	to split into_words
n_tok_inps	number tok inputs
...	additional arguments

### Details

as a byproduct of the tokenization process in the ‘encodes’ method.

### Value

None

---

HF-TokenClassInput	<i>HF-TokenClassInput</i>
--------------------	---------------------------

---

**Description**

HF-TokenClassInput

**Usage**

HF-TokenClassInput()

**Value**

None

---

HF-TokenTensorCategory	<i>HF-TokenTensorCategory</i>
------------------------	-------------------------------

---

**Description**

HF-TokenTensorCategory

**Usage**

HF-TokenTensorCategory()

**Value**

None

---

Hook	<i>Hook</i>
------	-------------

---

**Description**

Create a hook on 'm' with 'hook\_func'.

**Usage**

```
Hook(
    m,
    hook_func,
    is_forward = TRUE,
    detach = TRUE,
    cpu = FALSE,
    gather = FALSE
)
```

**Arguments**

m	m aparameter
hook_func	hook function
is_forward	is_forward or not
detach	detach or not
cpu	cpu or not
gather	gather or not

**Details**

Hooks are functions you can attach to a particular layer in your model and that will be executed in the forward pass (for forward hooks) or backward pass (for backward hooks).

**Value**

None

---

HookCallback

*HookCallback*

---

**Description**

‘Callback‘ that can be used to register hooks on ‘modules‘

‘Callback‘ that can be used to register hooks on ‘modules‘

**Usage**

```
HookCallback(
    modules = NULL,
    every = NULL,
    remove_end = TRUE,
    is_forward = TRUE,
    detach = TRUE,
    cpu = TRUE
)
```



```

)

HookCallback(
    modules = NULL,
    every = NULL,
    remove_end = TRUE,
    is_forward = TRUE,
    detach = TRUE,
    cpu = TRUE
)

```

**Arguments**

modules	modules
every	every
remove_end	remove_end or not
is_forward	is_forward or not
detach	detach or not
cpu	cpu or not

**Value**

None  
None

---

Hooks

*Hooks*

---

**Description**

Create several hooks on the modules in 'ms' with 'hook\_func'.

**Usage**

```
Hooks(ms, hook_func, is_forward = TRUE, detach = TRUE, cpu = FALSE)
```

**Arguments**

ms	ms parameter
hook_func	hook function
is_forward	is_forward or not
detach	detach or not
cpu	cpu or not

**Value**

None

---

hook_output	<i>Hook_output</i>
-------------	--------------------

---

**Description**

Return a 'Hook' that stores activations of 'module' in 'self\$.stored'

**Usage**

```
hook_output(module, detach = TRUE, cpu = FALSE, grad = FALSE)
```

**Arguments**

module	module
detach	detach or not
cpu	cpu or not
grad	grad or not

**Value**

None

---

hook_outputs	<i>Hook_outputs</i>
--------------	---------------------

---

**Description**

Return 'Hooks' that store activations of all 'modules' in 'self\$.stored'

**Usage**

```
hook_outputs(modules, detach = TRUE, cpu = FALSE, grad = FALSE)
```

**Arguments**

modules	modules
detach	detach or not
cpu	cpu or not
grad	grad or not

**Value**

None

---

hsv2rgb	<i>Hsv2rgb</i>
---------	----------------

---

**Description**

Converts a HSV image to an RGB image.

**Usage**

```
hsv2rgb(img)
```

**Arguments**

img	image object
-----	--------------

**Value**

None

---

Hue	<i>Hue</i>
-----	------------

---

**Description**

Apply change in hue of 'max\_hue' to batch of images with probability 'p'.

**Usage**

```
Hue(max_hue = 0.1, p = 0.75, draw = NULL, batch = FALSE)
```

**Arguments**

max_hue	maximum hue
p	probability
draw	draw
batch	batch

**Value**

None

hug                      *Transformer module*

---

**Description**

Transformer module

**Usage**

hug()

**Value**

None

---

icevision                *Icevision module*

---

**Description**

Icevision module

**Usage**

icevision()

**Value**

None

---

icevision\_Adapter      *Adapter*

---

**Description**

Adapter that enables the use of albumentations transforms.

**Usage**

icevision\_Adapter(tfms)

**Arguments**

tfms                      ‘Sequence’ of albumentation transforms.

**Value**

None

---

icevision\_aug\_tfms      *Aug\_tfms*

---

## Description

Collection of useful augmentation transforms.

## Usage

```
icevision_aug_tfms(
    size,
    presize = NULL,
    horizontal_flip = icevision_HorizontalFlip(always_apply = FALSE, p = 0.5),
    shift_scale_rotate = icevision_ShiftScaleRotate(always_apply = FALSE, p = 0.5,
        shift_limit_x = c(-0.0625, 0.0625), shift_limit_y = c(-0.0625, 0.0625), scale_limit =
        c(-0.1, 0.1), rotate_limit = c(-45, 45), interpolation = 1, border_mode = 4, value =
        NULL, mask_value = NULL),
    rgb_shift = icevision_RGBShift(always_apply = FALSE, p = 0.5, r_shift_limit = c(-20,
        20), g_shift_limit = c(-20, 20), b_shift_limit = c(-20, 20)),
    lightning = icevision_RandomBrightnessContrast(always_apply = FALSE, p = 0.5,
        brightness_limit = c(-0.2, 0.2), contrast_limit = c(-0.2, 0.2), brightness_by_max =
        TRUE),
    blur = icevision_Blur(always_apply = FALSE, p = 0.5, blur_limit = c(1, 3)),
    crop_fn = partial(icevision_RandomSizedBBBoxSafeCrop, p = 0.5),
    pad = partial(icevision_PadIfNeeded, border_mode = 0, value = list(124, 116, 104))
)
```

## Arguments

size	The final size of the image. If an 'int' is given, the maximum size of the image is rescaled, maintaing aspect ratio. If a 'list' is given, the image is rescaled to have that exact size (height, width).
presize	presize
horizontal_flip	Flip around the y-axis. If 'NULL' this transform is not applied.
shift_scale_rotate	Randomly shift, scale, and rotate. If 'NULL' this transform is not applied.
rgb_shift	Randomly shift values for each channel of RGB image. If 'NULL' this transform is not applied.
lightning	Randomly changes Brightness and Contrast. If 'NULL' this transform is not applied.
blur	Randomly blur the image. If 'NULL' this transform is not applied.
crop_fn	Randomly crop the image. If 'NULL' this transform is not applied. Use 'partial' to saturate other parameters of the class.
pad	Pad the image to 'size', squaring the image if 'size' is an 'int'. If 'NULL' this transform is not applied. Use 'partial' to saturate other parameters of the class.

**Value**

None

---

icevision\_BasicIAATransform  
*BasicIAATransform*

---

**Description**

BasicIAATransform

**Usage**

```
icevision_BasicIAATransform(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

---

icevision\_BasicTransform  
*BasicTransform*

---

**Description**

BasicTransform

**Usage**

```
icevision_BasicTransform(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

---

icevision_Blor	<i>Blor</i>
----------------	-------------

---

**Description**

Blor the input image using a random-sized kernel.

**Usage**

```
icevision_Blor(blur_limit = 7, always_apply = FALSE, p = 0.5)
```

**Arguments**

blur_limit	blur_limit
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

icevision_ChannelDropout	<i>ChannelDropout</i>
--------------------------	-----------------------

---

**Description**

Randomly Drop Channels in the input Image.

**Usage**

```
icevision_ChannelDropout(  
  channel_drop_range = list(1, 1),  
  fill_value = 0,  
  always_apply = FALSE,  
  p = 0.5  
)
```

**Arguments**

channel_drop_range	channel_drop_range
fill_value	fill_value
always_apply	always_apply
p	p

**Targets**

image

**Image types**

uint8, uint16, unit32, float32

---

icevision\_ChannelShuffle  
*ChannelShuffle*

---

**Description**

Randomly rearrange channels of the input RGB image.

**Usage**

```
icevision_ChannelShuffle(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32



---

icevision_CLAHE	<i>CLAHE</i>
-----------------	--------------

---

**Description**

Apply Contrast Limited Adaptive Histogram Equalization to the input image.

**Usage**

```
icevision_CLAHE(  
    clip_limit = 4,  
    tile_grid_size = list(8, 8),  
    always_apply = FALSE,  
    p = 0.5  
)
```

**Arguments**

clip_limit	clip_limit
tile_grid_size	tile_grid_size
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8

---

icevision_ClassMap	<i>ClassMap</i>
--------------------	-----------------

---

**Description**

Utility class for mapping between class name and id.

**Usage**

```
icevision_ClassMap(classes, background = 0)
```

**Arguments**

classes	classes
background	background

**Value**

Python dictionary

---

icevision\_CoarseDropout  
*CoarseDropout*

---

**Description**

CoarseDropout of the rectangular regions in the image.

**Usage**

```
icevision_CoarseDropout(
    max_holes = 8,
    max_height = 8,
    max_width = 8,
    min_holes = NULL,
    min_height = NULL,
    min_width = NULL,
    fill_value = 0,
    mask_fill_value = NULL,
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

max_holes	max_holes
max_height	max_height
max_width	max_width
min_holes	min_holes
min_height	min_height
min_width	min_width
fill_value	fill_value
mask_fill_value	mask_fill_value
always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask

**Image types**

uint8, float32

**Reference**

| <https://arxiv.org/abs/1708.04552> | <https://github.com/uoguelph-mlrg/Cutout/blob/master/util/cutout.py>  
| <https://github.com/aleju/imgaug/blob/master/imgaug/augmenters/arithmetic.py>

---

icevision\_ColorJitter *ColorJitter*

---

**Description**

Randomly changes the brightness, contrast, and saturation of an image. Compared to ColorJitter from torchvision,

**Usage**

```
icevision_ColorJitter(  
    brightness = 0.2,  
    contrast = 0.2,  
    saturation = 0.2,  
    hue = 0.2,  
    always_apply = False,  
    p = 0.5  
)
```

**Arguments**

brightness	brightness
contrast	contrast
saturation	saturation
hue	hue
always_apply	always_apply
p	p

**Details**

this transform gives a little bit different results because Pillow (used in torchvision) and OpenCV (used in Albumentations) transform an image to HSV format by different formulas. Another difference - Pillow uses uint8 overflow, but we use value saturation.

**Value**

None

---

icevision_Compose	<i>Compose</i>
-------------------	----------------

---

**Description**

Compose transforms and handle all transformations regarding bounding boxes

**Usage**

```
icevision_Compose(
    transforms,
    bbox_params = NULL,
    keypoint_params = NULL,
    additional_targets = NULL,
    p = 1
)
```

**Arguments**

transforms	transforms
bbox_params	bbox_params
keypoint_params	keypoint_params
additional_targets	additional_targets
p	p

**Value**

None

---

icevision_Crop	<i>Crop</i>
----------------	-------------

---

### Description

Crop region from image.

### Usage

```
icevision_Crop(
    x_min = 0,
    y_min = 0,
    x_max = 1024,
    y_max = 1024,
    always_apply = FALSE,
    p = 1
)
```

### Arguments

x_min	x_min
y_min	y_min
x_max	x_max
y_max	y_max
always_apply	always_apply
p	p

### Targets

image, mask, bboxes, keypoints

### Image types

uint8, float32

---

icevision_CropNonEmptyMaskIfExists	<i>CropNonEmptyMaskIfExists</i>
------------------------------------	---------------------------------

---

### Description

Crop area with mask if mask is non-empty, else make random crop.

**Usage**

```

icevision_CropNonEmptyMaskIfExists(
    height,
    width,
    ignore_values = NULL,
    ignore_channels = NULL,
    always_apply = FALSE,
    p = 1
)

```

**Arguments**

height	height
width	width
ignore_values	ignore_values
ignore_channels	ignore_channels
always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

icevision_Cutout	<i>Cutout</i>
------------------	---------------

---

**Description**

CoarseDropout of the square regions in the image.

**Usage**

```
icevision_Cutout(  
    num_holes = 8,  
    max_h_size = 8,  
    max_w_size = 8,  
    fill_value = 0,  
    always_apply = FALSE,  
    p = 0.5  
)
```

**Arguments**

num_holes	num_holes
max_h_size	max_h_size
max_w_size	max_w_size
fill_value	fill_value
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32

**Reference**

| <https://arxiv.org/abs/1708.04552> | <https://github.com/uoguelph-mlrg/Cutout/blob/master/util/cutout.py>  
| <https://github.com/aleju/imgaug/blob/master/imgaug/augmenters/arithmetic.py>

---

icevision_Dataset	<i>Dataset</i>
-------------------	----------------

---

**Description**

Container for a list of records and transforms.

**Usage**

```
icevision_Dataset(records, tfm = NULL)
```

**Arguments**

records	A list of records.
tfm	Transforms to be applied to each item.

**Details**

Steps each time an item is requested (normally via directly indexing the ‘Dataset’): Grab a record from the internal list of records. Prepare the record (open the image, open the mask, add metadata). Apply transforms to the record.

**Value**

None

---

icevision\_Dataset\_from\_images  
*Icevision Dataset from images*

---

**Description**

Creates a ‘Dataset’ from a list of images.

**Usage**

```
icevision_Dataset_from_images(images, tfm = NULL, ...)
```

**Arguments**

images	‘Sequence’ of images in memory (numpy arrays).
tfm	Transforms to be applied to each item.
...	additional arguments

**Value**

None



---

icevision\_Downscale    *Downscale*

---

**Description**

Decreases image quality by downscaling and upscaling back.

**Usage**

```
icevision_Downscale(  
    scale_min = 0.25,  
    scale_max = 0.25,  
    interpolation = 0,  
    always_apply = FALSE,  
    p = 0.5  
)
```

**Arguments**

scale_min	scale_min
scale_max	scale_max
interpolation	cv2 interpolation method. cv2.INTER_NEAREST by default
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32

icevision\_DualIAATransform  
*DualIAATransform*

---

**Description**

Transform for segmentation task.

**Usage**

```
icevision_DualIAATransform(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

---

icevision\_DualTransform  
*DualTransform*

---

**Description**

Transform for segmentation task.

**Usage**

```
icevision_DualTransform(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

---

icevision\_ElasticTransform  
*ElasticTransform*

---

### Description

Elastic deformation of images as described in [Simard2003]\_ (with modifications).

### Usage

```
icevision_ElasticTransform(  
    alpha = 1,  
    sigma = 50,  
    alpha_affine = 50,  
    interpolation = 1,  
    border_mode = 4,  
    value = NULL,  
    mask_value = NULL,  
    always_apply = FALSE,  
    approximate = FALSE,  
    p = 0.5  
)
```

### Arguments

alpha	alpha
sigma	sigma
alpha_affine	alpha_affine
interpolation	interpolation
border_mode	border_mode
value	value
mask_value	mask_value
always_apply	always_apply
approximate	approximate
p	p

### Details

Based on <https://gist.github.com/erniejunior/601cdf56d2b424757de5> .. [Simard2003] Simard, Steinkraus and Platt, "Best Practices for Convolutional Neural Networks applied to Visual Document Analysis", in Proc. of the International Conference on Document Analysis and Recognition, 2003.

### Value

None

**Targets**

image, mask

**Image types**

uint8, float32

---

icevision\_Equalize     *Equalize*

---

**Description**

Equalize the image histogram.

**Usage**

```
icevision_Equalize(mode = "cv", by_channels = TRUE, mask = NULL, ...)
```

**Arguments**

mode	mode
by_channels	by_channels
mask	mask
...	additional arguments

**Value**

None

**Targets**

image

**Image types**

uint8

---

icevision_FancyPCA	<i>FancyPCA</i>
--------------------	-----------------

---

**Description**

Augment RGB image using FancyPCA from Krizhevsky's paper

**Usage**

```
icevision_FancyPCA(alpha = 0.1, always_apply = FALSE, p = 0.5)
```

**Arguments**

alpha	alpha
always_apply	always_apply
p	p

**Details**

"ImageNet Classification with Deep Convolutional Neural Networks"

**Value**

None

**Targets**

image

**Image types**

3-channel uint8 images only

**Credit**

<http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>  
<https://deshanadesai.github.io/notes/Fancy-PCA-with-Scikit-Image> [https://pixelatedbrian.github.io/2018-04-29-fancy\\_pca/](https://pixelatedbrian.github.io/2018-04-29-fancy_pca/)

---

icevision_FDA	FDA
---------------	-----

---

## Description

Fourier Domain Adaptation from <https://github.com/YanchaoYang/FDA>

## Usage

```
icevision_FDA(  
    reference_images,  
    beta_limit = 0.1,  
    read_fn = icevision_read_rgb_image(),  
    always_apply = FALSE,  
    p = 0.5  
)
```

## Arguments

reference_images	reference_images
beta_limit	beta_limit
read_fn	read_fn
always_apply	always_apply
p	p

## Details

Simple "style transfer".

## Value

None

## Fourier Domain Adaptation from [https](https://github.com/YanchaoYang/FDA)

[//github.com/YanchaoYang/FDA](https://github.com/YanchaoYang/FDA): Simple "style transfer".

## Targets

image

## Image types

uint8, float32

**Reference**

<https://github.com/YanchaoYang/FDA> [https://openaccess.thecvf.com/content\\_CVPR\\_2020/papers/Yang\\_FDA\\_Fourier\\_Dom](https://openaccess.thecvf.com/content_CVPR_2020/papers/Yang_FDA_Fourier_Dom)

**Example**

```
>> import numpy as np >> import albumentations as A >> image = np.random.randint(0, 256, [100,
100, 3], dtype=np.uint8) >> target_image = np.random.randint(0, 256, [100, 100, 3], dtype=np.uint8)
>> aug = A.Compose([A.FDA([target_image], p=1, read_fn=lambda x: x)]) >> result = aug(image=image)
```

---

icevision\_FixedSplitter  
*FixedSplitter*

---

**Description**

Split 'ids' based on predefined splits.

**Usage**

```
icevision_FixedSplitter(splits)
```

**Arguments**

splits            The predefined splits.

**Value**

None

---

icevision\_Flip            *Flip*

---

**Description**

Flip the input either horizontally, vertically or both horizontally and vertically.

**Usage**

```
icevision_Flip(always_apply = FALSE, p = 0.5)
```

**Arguments**

always\_apply    always\_apply  
p                p

**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

 icevision\_FromFloat *FromFloat*


---

**Description**

Take an input array where all values should lie in the range [0, 1.0], multiply them by ‘max\_value’ and then

**Usage**

```
icevision_FromFloat(
    dtype = "uint16",
    max_value = NULL,
    always_apply = FALSE,
    p = 1
)
```

**Arguments**

dtype	dtype
max_value	max_value
always_apply	always_apply
p	p

**Details**

cast the resulted value to a type specified by ‘dtype’. If ‘max\_value’ is NULL the transform will try to infer the maximum value for the data type from the ‘dtype’ argument. This is the inverse transform for :class:~alumentations.augmentations.transforms.ToFloat’.

**Value**

None

**Targets**

image



**Image types**

float32

---

icevision\_GaussianBlur  
*GaussianBlur*

---

**Description**

Blur the input image using a Gaussian filter with a random kernel size.

**Usage**

```
icevision_GaussianBlur(  
    blur_limit = list(3, 7),  
    sigma_limit = 0,  
    always_apply = FALSE,  
    p = 0.5  
)
```

**Arguments**

blur_limit	blur_limit
sigma_limit	sigma_limit
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

icevision\_GaussNoise    *GaussNoise*

---

**Description**

Apply gaussian noise to the input image.

**Usage**

```
icevision_GaussNoise(  
    var_limit = list(10, 50),  
    mean = 0,  
    always_apply = FALSE,  
    p = 0.5  
)
```

**Arguments**

var_limit	var_limit
mean	mean
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

icevision\_GlassBlur    *GlassBlur*

---

**Description**

Apply glass noise to the input image.

**Usage**

```
icevision_GlassBlur(  
    sigma = 0.7,  
    max_delta = 4,  
    iterations = 2,  
    always_apply = False,  
    mode = "fast",  
    p = 0.5  
)
```

**Arguments**

sigma	sigma
max_delta	max_delta
iterations	iterations
always_apply	always_apply
mode	mode
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32

**Reference**

| <https://arxiv.org/abs/1903.12261> | [https://github.com/hendrycks/robustness/blob/master/ImageNet-C/create\\_c/make\\_imagenet\\_c.py](https://github.com/hendrycks/robustness/blob/master/ImageNet-C/create_c/make_imagenet_c.py)

---

icevision\_GridDistortion

*GridDistortion*

---

**Description**

Args:

**Usage**

```
icevision_GridDistortion(
    num_steps = 5,
    distort_limit = 0.3,
    interpolation = 1,
    border_mode = 4,
    value = NULL,
    mask_value = NULL,
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

num_steps	num_steps
distort_limit	distort_limit
interpolation	interpolation
border_mode	border_mode
value	value
mask_value	mask_value
always_apply	always_apply
p	p

**Details**

num\_steps (int): count of grid cells on each side. distort\_limit (float, (float, float)): If distort\_limit is a single float, the range will be (-distort\_limit, distort\_limit). Default: (-0.03, 0.03). interpolation (OpenCV flag): flag that is used to specify the interpolation algorithm. Should be one of: cv2.INTER\_NEAREST, cv2.INTER\_LINEAR, cv2.INTER\_CUBIC, cv2.INTER\_AREA, cv2.INTER\_LANCZOS4. Default: cv2.INTER\_LINEAR. border\_mode (OpenCV flag): flag that is used to specify the pixel extrapolation method. Should be one of: cv2.BORDER\_CONSTANT, cv2.BORDER\_REPLICATE, cv2.BORDER\_REFLECT, cv2.BORDER\_WRAP, cv2.BORDER\_REFLECT\_101. Default: cv2.BORDER\_REFLECT\_101. value (int, float, list of ints, list of float): padding value if border\_mode is cv2.BORDER\_CONSTANT. mask\_value (int, float, list of ints, list of float): padding value if border\_mode is cv2.BORDER\_CONSTANT applied for masks. Targets: image, mask Image types: uint8, float32

**Value**

None

**Targets**

image, mask

**Image types**

uint8, float32

---

 icevision\_GridDropout *GridDropout*


---

**Description**

GridDropout, drops out rectangular regions of an image and the corresponding mask in a grid fashion.

**Usage**

```
icevision_GridDropout(
    ratio = 0.5,
    unit_size_min = NULL,
    unit_size_max = NULL,
    holes_number_x = NULL,
    holes_number_y = NULL,
    shift_x = 0,
    shift_y = 0,
    random_offset = FALSE,
    fill_value = 0,
    mask_fill_value = NULL,
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

ratio	ratio
unit_size_min	unit_size_min
unit_size_max	unit_size_max
holes_number_x	holes_number_x
holes_number_y	holes_number_y
shift_x	shift_x
shift_y	shift_y
random_offset	random_offset
fill_value	fill_value
mask_fill_value	mask_fill_value
always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask

**Image types**

uint8, float32

**References**

<https://arxiv.org/abs/2001.04086>

---

icevision\_HistogramMatching

*HistogramMatching*

---

**Description**

Apply histogram matching. It manipulates the pixels of an input image so that its histogram matches

**Usage**

```
icevision_HistogramMatching(  
    reference_images,  
    blend_ratio = list(0.5, 1),  
    read_fn = icevision_read_rgb_image(),  
    always_apply = FALSE,  
    p = 0.5  
)
```

**Arguments**

reference_images	reference_images
blend_ratio	blend_ratio
read_fn	read_fn
always_apply	always_apply
p	p

**Details**

the histogram of the reference image. If the images have multiple channels, the matching is done independently for each channel, as long as the number of channels is equal in the input image and the reference. Histogram matching can be used as a lightweight normalisation for image processing, such as feature matching, especially in circumstances where the images have been taken from different sources or in different conditions (i.e. lighting). See: [https://scikit-image.org/docs/dev/auto\\_examples/color\\_exposure/plot\\_histogram\\_matching.html](https://scikit-image.org/docs/dev/auto_examples/color_exposure/plot_histogram_matching.html)

**Value**

None

**See**

[https://scikit-image.org/docs/dev/auto\\_examples/color\\_exposure/plot\\_histogram\\_matching.html](https://scikit-image.org/docs/dev/auto_examples/color_exposure/plot_histogram_matching.html)

**Targets**

image

**Image types**

uint8, uint16, float32

---

icevision\_HorizontalFlip  
*HorizontalFlip*

---

**Description**

Flip the input horizontally around the y-axis.

**Usage**

```
icevision_HorizontalFlip(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

`icevision_HueSaturationValue`*HueSaturationValue*

---

**Description**

Randomly change hue, saturation and value of the input image.

**Usage**

```
icevision_HueSaturationValue(  
    hue_shift_limit = 20,  
    sat_shift_limit = 30,  
    val_shift_limit = 20,  
    always_apply = FALSE,  
    p = 0.5  
)
```

**Arguments**

<code>hue_shift_limit</code>	<code>hue_shift_limit</code>
<code>sat_shift_limit</code>	<code>sat_shift_limit</code>
<code>val_shift_limit</code>	<code>val_shift_limit</code>
<code>always_apply</code>	<code>always_apply</code>
<code>p</code>	<code>p</code>

**Value**

None

**Targets**

image

**Image types**

uint8, float32



---

icevision\_IAAAdditiveGaussianNoise  
*IAAAdditiveGaussianNoise*

---

**Description**

Add gaussian noise to the input image.

**Usage**

```
icevision_IAAAdditiveGaussianNoise(  
    loc = 0,  
    scale = list(2.55, 12.75),  
    per_channel = FALSE,  
    always_apply = FALSE,  
    p = 0.5  
)
```

**Arguments**

loc	loc
scale	scale
per_channel	per_channel
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

---

icevision\_IAAAffine *IAAAAffine*

---

**Description**

Place a regular grid of points on the input and randomly move the neighbourhood of these point around

**Usage**

```
icevision_IAAAffine(
    scale = 1,
    translate_percent = NULL,
    translate_px = NULL,
    rotate = 0,
    shear = 0,
    order = 1,
    cval = 0,
    mode = "reflect",
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

scale	scale
translate_percent	translate_percent
translate_px	translate_px
rotate	rotate
shear	shear
order	order
cval	cval
mode	mode
always_apply	always_apply
p	p

**Details**

via affine transformations. Note: This class introduce interpolation artifacts to mask if it has values other than 0;1

**Value**

None

None

**Targets**

image, mask

---

icevision\_IACropAndPad  
*IACropAndPad*

---

**Description**

Transform for segmentation task.

**Usage**

```
icevision_IACropAndPad(  
    px = NULL,  
    percent = NULL,  
    pad_mode = "constant",  
    pad_cval = 0,  
    keep_size = TRUE,  
    always_apply = FALSE,  
    p = 1  
)
```

**Arguments**

px	px
percent	percent
pad_mode	pad_mode
pad_cval	pad_cval
keep_size	keep_size
always_apply	always_apply
p	p

---

icevision\_IAAmboss    *IAAmboss*

---

**Description**

Emboss the input image and overlays the result with the original image.

**Usage**

```
icevision_IAAmboss(  
    alpha = list(0.2, 0.5),  
    strength = list(0.2, 0.7),  
    always_apply = FALSE,  
    p = 0.5  
)
```

**Arguments**

alpha	alpha
strength	strength
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

---

icevision\_IAAFliplr    *IAAFliplr*

---

**Description**

Transform for segmentation task.

**Usage**

```
icevision_IAAFliplr(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

---

icevision\_IAAFlipud    *IAAFlipud*

---

**Description**

Transform for segmentation task.

**Usage**

```
icevision_IAAFlipud(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

---

icevision\_IAAPerspective  
*IAAPerspective*

---

**Description**

Perform a random four point perspective transform of the input.

**Usage**

```
icevision_IAAPerspective(  
  scale = list(0.05, 0.1),  
  keep_size = TRUE,  
  always_apply = FALSE,  
  p = 0.5  
)
```

**Arguments**

scale	scale
keep_size	keep_size
always_apply	always_apply
p	p

**Details**

Note: This class introduce interpolation artifacts to mask if it has values other than 0;1

**Value**

None

**Targets**

image, mask

---

icevision\_IAPiecewiseAffine  
*IAPiecewiseAffine*

---

**Description**

Place a regular grid of points on the input and randomly move the neighbourhood of these point around

**Usage**

```
icevision_IAPiecewiseAffine(
    scale = list(0.03, 0.05),
    nb_rows = 4,
    nb_cols = 4,
    order = 1,
    cval = 0,
    mode = "constant",
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

scale	scale
nb_rows	nb_rows
nb_cols	nb_cols
order	order
cval	cval
mode	mode
always_apply	always_apply
p	p

**Details**

via affine transformations. Note: This class introduces interpolation artifacts to mask if it has values other than 0;1

**Value**

None

**Targets**

image, mask

---

icevision\_IAASharpener *IAASharpener*

---

**Description**

Sharpen the input image and overlays the result with the original image.

**Usage**

```
icevision_IAASharpener(  
    alpha = list(0.2, 0.5),  
    lightness = list(0.5, 1),  
    always_apply = FALSE,  
    p = 0.5  
)
```

**Arguments**

alpha	alpha
lightness	lightness
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

---

icevision\_IAASuperpixels  
*IAASuperpixels*

---

### Description

Completely or partially transform the input image to its superpixel representation. Uses skimage's version

### Usage

```
icevision_IAASuperpixels(  
    p_replace = 0.1,  
    n_segments = 100,  
    always_apply = FALSE,  
    p = 0.5  
)
```

### Arguments

p_replace	p_replace
n_segments	n_segments
always_apply	always_apply
p	p

### Details

of the SLIC algorithm. May be slow.

### Value

None

### Targets

image



---

icevision\_ImageCompression  
*ImageCompression*

---

## Description

Decrease Jpeg, WebP compression of an image.

## Usage

```
icevision_ImageCompression(  
    quality_lower = 99,  
    quality_upper = 100,  
    compression_type = 0,  
    always_apply = FALSE,  
    p = 0.5  
)
```

## Arguments

quality_lower	quality_lower
quality_upper	quality_upper
compression_type	compression_type
always_apply	always_apply
p	p

## Value

None

## Targets

image

## Image types

uint8, float32

icevision\_ImageOnlyIAATransform  
*ImageOnlyIAATransform*

---

**Description**

Transform applied to image only.

**Usage**

```
icevision_ImageOnlyIAATransform(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

---

icevision\_ImageOnlyTransform  
*ImageOnlyTransform*

---

**Description**

Transform applied to image only.

**Usage**

```
icevision_ImageOnlyTransform(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

---

icevision\_InvertImg     *InvertImg*

---

**Description**

Invert the input image by subtracting pixel values from 255.

**Usage**

```
icevision_InvertImg(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8

---

icevision\_ISONoise     *ISONoise*

---

**Description**

Apply camera sensor noise.

**Usage**

```
icevision_ISONoise(  
  color_shift = list(0.01, 0.05),  
  intensity = list(0.1, 0.5),  
  always_apply = FALSE,  
  p = 0.5  
)
```

**Arguments**

color_shift	color_shift
intensity	intensity
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8

---

icevision\_JpegCompression  
*JpegCompression*

---

**Description**

Decrease Jpeg compression of an image.

**Usage**

```
icevision_JpegCompression(
    quality_lower = 99,
    quality_upper = 100,
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

quality_lower	quality_lower
quality_upper	quality_upper
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

icevision\_LongestMaxSize  
*LongestMaxSize*

---

**Description**

Rescale an image so that maximum side is equal to max\_size, keeping the aspect ratio of the initial image.

**Usage**

```
icevision_LongestMaxSize(  
    max_size = 1024,  
    interpolation = 1,  
    always_apply = FALSE,  
    p = 1  
)
```

**Arguments**

max_size	max_size
interpolation	interpolation
always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

icevision\_MaskDropout *MaskDropout*

---

### Description

Image & mask augmentation that zero out mask and image regions corresponding

### Usage

```
icevision_MaskDropout(  
    max_objects = 1,  
    image_fill_value = 0,  
    mask_fill_value = 0,  
    always_apply = FALSE,  
    p = 0.5  
)
```

### Arguments

max_objects	max_objects
image_fill_value	image_fill_value
mask_fill_value	mask_fill_value
always_apply	always_apply
p	p

### Details

to randomly chosen object instance from mask. Mask must be single-channel image, zero values treated as background. Image can be any number of channels. Inspired by <https://www.kaggle.com/c/severstal-steel-defect-detection/discussion/114254>

### Value

None

---

icevision\_MedianBlur *MedianBlur*

---

**Description**

Blur the input image using a median filter with a random aperture linear size.

**Usage**

```
icevision_MedianBlur(blur_limit = 7, always_apply = FALSE, p = 0.5)
```

**Arguments**

blur_limit	blur_limit
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

icevision\_MotionBlur *MotionBlur*

---

**Description**

Apply motion blur to the input image using a random-sized kernel.

**Usage**

```
icevision_MotionBlur(blur_limit = 7, always_apply = FALSE, p = 0.5)
```

**Arguments**

blur_limit	blur_limit
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

 icevision\_MultiplicativeNoise  
*MultiplicativeNoise*


---

**Description**

Multiply image to random number or array of numbers.

**Usage**

```
icevision_MultiplicativeNoise(
    multiplier = list(0.9, 1.1),
    per_channel = FALSE,
    elementwise = FALSE,
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

multiplier	multiplier
per_channel	per_channel
elementwise	elementwise
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

Any



---

icevision\_Normalize    *Normalize*

---

### Description

Divide pixel values by  $255 = 2^{*}8 - 1$ , subtract mean per channel and divide by std per channel.

### Usage

```
icevision_Normalize(  
    mean = list(0.485, 0.456, 0.406),  
    std = list(0.229, 0.224, 0.225),  
    max_pixel_value = 255,  
    always_apply = FALSE,  
    p = 1  
)
```

### Arguments

mean	mean
std	std
max_pixel_value	max_pixel_value
always_apply	always_apply
p	p

### Value

None

### Targets

image

### Image types

uint8, float32

---

```
icevision_OpticalDistortion
    OpticalDistortion
```

---

## Description

OpticalDistortion

## Usage

```
icevision_OpticalDistortion(
    distort_limit = 0.05,
    shift_limit = 0.05,
    interpolation = 1,
    border_mode = 4,
    value = NULL,
    mask_value = NULL,
    always_apply = FALSE,
    p = 0.5
)
```

## Arguments

<code>distort_limit</code>	<code>distort_limit</code>
<code>shift_limit</code>	<code>shift_limit</code>
<code>interpolation</code>	<code>interpolation</code>
<code>border_mode</code>	<code>border_mode</code>
<code>value</code>	<code>value</code>
<code>mask_value</code>	<code>mask_value</code>
<code>always_apply</code>	<code>always_apply</code>
<code>p</code>	<code>p</code>

## Details

`distort_limit` (float, (float, float)): If `distort_limit` is a single float, the range will be  $(-distort\_limit, distort\_limit)$ . Default:  $(-0.05, 0.05)$ . `shift_limit` (float, (float, float)): If `shift_limit` is a single float, the range will be  $(-shift\_limit, shift\_limit)$ . Default:  $(-0.05, 0.05)$ . `interpolation` (OpenCV flag): flag that is used to specify the interpolation algorithm. Should be one of: `cv2.INTER_NEAREST`, `cv2.INTER_LINEAR`, `cv2.INTER_CUBIC`, `cv2.INTER_AREA`, `cv2.INTER_LANCZOS4`. Default: `cv2.INTER_LINEAR`. `border_mode` (OpenCV flag): flag that is used to specify the pixel extrapolation method. Should be one of: `cv2.BORDER_CONSTANT`, `cv2.BORDER_REPLICATE`, `cv2.BORDER_REFLECT`, `cv2.BORDER_WRAP`, `cv2.BORDER_REFLECT_101`. Default: `cv2.BORDER_REFLECT_101`. `value` (int, float, list of ints, list of float): padding value if `border_mode` is `cv2.BORDER_CONSTANT`. `mask_value` (int, float, list of ints, list of float): padding value if `border_mode` is `cv2.BORDER_CONSTANT` applied for masks. Targets: image, mask Image types: uint8, float32

**Value**

None

**Targets**

image, mask

**Image types**

uint8, float32

---

 icevision\_PadIfNeeded *PadIfNeeded*


---

**Description**

Pad side of the image / max if side is less than desired number.

**Usage**

```
icevision_PadIfNeeded(
    min_height = 1024,
    min_width = 1024,
    pad_height_divisor = NULL,
    pad_width_divisor = NULL,
    border_mode = 4,
    value = NULL,
    mask_value = NULL,
    always_apply = FALSE,
    p = 1
)
```

**Arguments**

min_height	min_height
min_width	min_width
pad_height_divisor	pad_height_divisor
pad_width_divisor	pad_width_divisor
border_mode	border_mode
value	value
mask_value	mask_value
always_apply	always_apply
p	p

**Targets**

image, mask, bbox, keypoints

**Image types**

uint8, float32

---

icevision_parse	<i>Parse</i>
-----------------	--------------

---

**Description**

Loops through all data points parsing the required fields.

**Usage**

```
icevision_parse(  
    data_splitter = NULL,  
    idmap = NULL,  
    autofix = TRUE,  
    show_pbar = TRUE,  
    cache_filepath = NULL  
)
```

**Arguments**

<code>data_splitter</code>	How to split the parsed data, defaults to a [0.8, 0.2] random split.
<code>idmap</code>	Maps from filenames to unique ids, pass an 'IDMap()' if you need this information.
<code>autofix</code>	autofix
<code>show_pbar</code>	Whether or not to show a progress bar while parsing the data.
<code>cache_filepath</code>	Path to save records in pickle format. Defaults to NULL, e.g. if the user does not specify a path, no saving nor loading happens.

**Value**

A list of records for each split defined by `data_splitter`.

---

icevision\_Posterize     *Posterize*

---

**Description**

Reduce the number of bits for each color channel.

**Usage**

```
icevision_Posterize(num_bits = 4, always_apply = FALSE, p = 0.5)
```

**Arguments**

num_bits	num_bits
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8

---

icevision\_RandomBrightnessContrast  
*RandomBrightnessContrast*

---

**Description**

Randomly change brightness and contrast of the input image.

**Usage**

```
icevision_RandomBrightnessContrast(  
  brightness_limit = 0.2,  
  contrast_limit = 0.2,  
  brightness_by_max = TRUE,  
  always_apply = FALSE,  
  p = 0.5  
)
```

**Arguments**

brightness_limit	brightness_limit
contrast_limit	contrast_limit
brightness_by_max	brightness_by_max
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

icevision\_RandomContrast

*RandomContrast*

---

**Description**

Randomly change contrast of the input image.

**Usage**

```
icevision_RandomContrast(limit = 0.2, always_apply = FALSE, p = 0.5)
```

**Arguments**

limit	limit
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

icevision\_RandomCrop *RandomCrop*

---

**Description**

Crop a random part of the input.

**Usage**

```
icevision_RandomCrop(height, width, always_apply = FALSE, p = 1)
```

**Arguments**

height	height
width	width
always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

```
icevision_RandomCropNearBBox
    RandomCropNearBBox
```

---

**Description**

Crop bbox from image with random shift by x,y coordinates

**Usage**

```
icevision_RandomCropNearBBox(max_part_shift = 0.3, always_apply = FALSE, p = 1)
```

**Arguments**

```
max_part_shift  max_part_shift
always_apply    always_apply
p               p
```

**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

```
icevision_RandomFog    RandomFog
```

---

**Description**

Simulates fog for the image

**Usage**

```
icevision_RandomFog(
  fog_coef_lower = 0.3,
  fog_coef_upper = 1,
  alpha_coef = 0.08,
  always_apply = FALSE,
  p = 0.5
)
```



**Arguments**

fog_coef_lower	fog_coef_lower
fog_coef_upper	fog_coef_upper
alpha_coef	alpha_coef
always_apply	always_apply
p	p

**Details**

From <https://github.com/UjjwalSaxena/Automold-Road-Augmentation-Library>

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

icevision\_RandomGamma *RandomGamma*

---

**Description**

RandomGamma

**Usage**

```
icevision_RandomGamma(
    gamma_limit = list(80, 120),
    eps = NULL,
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

gamma_limit	gamma_limit
eps	Deprecated.
always_apply	always_apply
p	p

**Details**

gamma\_limit (float or (float, float)): If gamma\_limit is a single float value, the range will be (-gamma\_limit, gamma\_limit). Default: (80, 120). eps: Deprecated. Targets: image Image types: uint8, float32

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

icevision\_RandomGridShuffle

*RandomGridShuffle*

---

**Description**

Random shuffle grid's cells on image.

**Usage**

```
icevision_RandomGridShuffle(grid = list(3, 3), always_apply = FALSE, p = 0.5)
```

**Arguments**

grid	grid
always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask

**Image types**

uint8, float32

---

 icevision\_RandomRain *RandomRain*


---

**Description**

Adds rain effects.

**Usage**

```
icevision_RandomRain(
    slant_lower = -10,
    slant_upper = 10,
    drop_length = 20,
    drop_width = 1,
    drop_color = list(200, 200, 200),
    blur_value = 7,
    brightness_coefficient = 0.7,
    rain_type = NULL,
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

slant_lower	should be in range [-20, 20].
slant_upper	should be in range [-20, 20].
drop_length	should be in range [0, 100].
drop_width	should be in range [1, 5]. drop_color (list of (r, g, b)): rain lines color. blur_value (int): rainy view are blurry brightness_coefficient (float): rainy days are usually shady. Should be in range [0, 1].
drop_color	drop_color
blur_value	blur_value
brightness_coefficient	brightness_coefficient
rain_type	One of [NULL, "drizzle", "heavy", "torrestial"]
always_apply	always_apply
p	p

**Details**

From <https://github.com/UjjwalSaxena/Automold-Road-Augmentation-Library>

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

```
icevision_RandomResizedCrop
    RandomResizedCrop
```

---

**Description**

Torchvision's variant of crop a random part of the input and rescale it to some size.

**Usage**

```
icevision_RandomResizedCrop(
    height,
    width,
    scale = list(0.08, 1),
    ratio = list(0.75, 1.3333333333333333),
    interpolation = 1,
    always_apply = FALSE,
    p = 1
)
```

**Arguments**

height	height
width	width
scale	scale
ratio	ratio
interpolation	interpolation
always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

icevision\_RandomRotate90  
*RandomRotate90*

---

**Description**

Randomly rotate the input by 90 degrees zero or more times.

**Usage**

```
icevision_RandomRotate90(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

icevision\_RandomScale *RandomScale*

---

**Description**

Randomly resize the input. Output image size is different from the input image size.

**Usage**

```
icevision_RandomScale(  
  scale_limit = 0.1,  
  interpolation = 1L,  
  always_apply = FALSE,  
  p = 0.5  
)
```

**Arguments**

scale_limit	scale_limit
interpolation	interpolation
always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

icevision\_RandomShadow

*RandomShadow*

---

**Description**

Simulates shadows for the image

**Usage**

```
icevision_RandomShadow(
    shadow_roi = list(0, 0.5, 1, 1),
    num_shadows_lower = 1,
    num_shadows_upper = 2,
    shadow_dimension = 5,
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

shadow_roi	shadow_roi
num_shadows_lower	num_shadows_lower
num_shadows_upper	num_shadows_upper
shadow_dimension	shadow_dimension
always_apply	always_apply
p	p

**Details**

From <https://github.com/UjjwalSaxena/Automold-Road-Augmentation-Library>

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

icevision\_RandomSizedBBoxSafeCrop  
*RandomSizedBBoxSafeCrop*

---

**Description**

Crop a random part of the input and rescale it to some size without loss of bboxes.

Crop a random part of the input and rescale it to some size without loss of bboxes.

**Usage**

```
icevision_RandomSizedBBoxSafeCrop(  
    height,  
    width,  
    erosion_rate = 0,  
    interpolation = 1,  
    always_apply = FALSE,  
    p = 1  
)
```

```
icevision_RandomSizedBBoxSafeCrop(  
    height,  
    width,  
    erosion_rate = 0,  
    interpolation = 1,  
    always_apply = FALSE,  
    p = 1  
)
```

**Arguments**

height	height
width	width
erosion_rate	erosion_rate
interpolation	interpolation
always_apply	always_apply
p	p

**Value**

None  
None

**Targets**

image, mask, bboxes  
image, mask, bboxes

**Image types**

uint8, float32  
uint8, float32

---

icevision\_RandomSizedCrop  
*RandomSizedCrop*

---

**Description**

Crop a random part of the input and rescale it to some size.

**Usage**

```
icevision_RandomSizedCrop(  
    min_max_height,  
    height,  
    width,  
    w2h_ratio = 1,  
    interpolation = 1,  
    always_apply = FALSE,  
    p = 1  
)
```



**Arguments**

min_max_height	min_max_height
height	height
width	width
w2h_ratio	w2h_ratio
interpolation	interpolation
always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

icevision\_RandomSnow *RandomSnow*

---

**Description**

Bleach out some pixel values simulating snow.

**Usage**

```
icevision_RandomSnow(
    snow_point_lower = 0.1,
    snow_point_upper = 0.3,
    brightness_coeff = 2.5,
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

snow_point_lower	snow_point_lower
snow_point_upper	snow_point_upper
brightness_coeff	brightness_coeff
always_apply	always_apply
p	p

**Details**

From <https://github.com/UjjwalSaxena/Automold-Road-Augmentation-Library>

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

icevision\_RandomSplitter  
*RandomSplitter*

---

**Description**

Randomly splits items.

**Usage**

```
icevision_RandomSplitter(probs, seed = NULL)
```

**Arguments**

probs	'Sequence' of probabilities that must sum to one. The length of the 'Sequence' is the number of groups to to split the items into.
seed	Internal seed used for shuffling the items. Define this if you need reproducible results.

**Value**

None

---

icevision\_RandomSunFlare  
*RandomSunFlare*

---

**Description**

Simulates Sun Flare for the image

**Usage**

```
icevision_RandomSunFlare(  
    flare_roi = list(0, 0, 1, 0.5),  
    angle_lower = 0,  
    angle_upper = 1,  
    num_flare_circles_lower = 6,  
    num_flare_circles_upper = 10,  
    src_radius = 400,  
    src_color = list(255, 255, 255),  
    always_apply = FALSE,  
    p = 0.5  
)
```

**Arguments**

flare_roi	flare_roi
angle_lower	angle_lower
angle_upper	angle_upper
num_flare_circles_lower	num_flare_circles_lower
num_flare_circles_upper	num_flare_circles_upper
src_radius	src_radius
src_color	src_color
always_apply	always_apply
p	p

**Details**

From <https://github.com/UjjwalSaxena/Automold-Road-Augmentation-Library>

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

`icevision_read_bgr_image`*Read\_bgr\_image*

---

**Description**

Read\_bgr\_image

**Usage**`icevision_read_bgr_image(path)`**Arguments**

path            path

**Value**

None

---

`icevision_read_rgb_image`*Read\_rgb\_image*

---

**Description**

Read\_rgb\_image

**Usage**`icevision_read_rgb_image(path)`**Arguments**

path            path

**Value**

None

---

icevision_Resize	<i>Resize</i>
------------------	---------------

---

**Description**

Resize the input to the given height and width.

**Usage**

```
icevision_Resize(height, width, interpolation = 1, always_apply = FALSE, p = 1)
```

**Arguments**

height	height
width	width
interpolation	interpolation
always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

icevision_resize_and_pad	<i>Resize_and_pad</i>
--------------------------	-----------------------

---

**Description**

Resize\_and\_pad

**Usage**

```
icevision_resize_and_pad(  
    size,  
    pad = partial(icevision_PadIfNeeded, border_mode = 0, value = c(124L, 116L, 104L))  
)
```

**Arguments**

size	size
pad	pad

**Value**

None

---

icevision_RGBShift	<i>RGBShift</i>
--------------------	-----------------

---

**Description**

Randomly shift values for each channel of the input RGB image.

Randomly shift values for each channel of the input RGB image.

**Usage**

```
icevision_RGBShift(
    r_shift_limit = 20,
    g_shift_limit = 20,
    b_shift_limit = 20,
    always_apply = FALSE,
    p = 0.5
)
```

```
icevision_RGBShift(
    r_shift_limit = 20,
    g_shift_limit = 20,
    b_shift_limit = 20,
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

r_shift_limit	r_shift_limit
g_shift_limit	g_shift_limit
b_shift_limit	b_shift_limit
always_apply	always_apply
p	p

**Value**

None

None

**Targets**

image  
image

**Image types**

uint8, float32  
uint8, float32

---

icevision_Rotate	<i>Rotate</i>
------------------	---------------

---

**Description**

Rotate the input by an angle selected randomly from the uniform distribution.

**Usage**

```
icevision_Rotate(
    limit = 90,
    interpolation = 1,
    border_mode = 4,
    value = NULL,
    mask_value = NULL,
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

limit	limit
interpolation	interpolation
border_mode	border_mode
value	value
mask_value	mask_value
always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

`icevision_ShiftScaleRotate`*ShiftScaleRotate*

---

**Description**

Randomly apply affine transforms: translate, scale and rotate the input.

Randomly apply affine transforms: translate, scale and rotate the input.

**Usage**

```
icevision_ShiftScaleRotate(
    shift_limit = 0.0625,
    scale_limit = 0.1,
    rotate_limit = 45,
    interpolation = 1,
    border_mode = 4,
    value = NULL,
    mask_value = NULL,
    shift_limit_x = NULL,
    shift_limit_y = NULL,
    always_apply = FALSE,
    p = 0.5
)
```

```
icevision_ShiftScaleRotate(
    shift_limit = 0.0625,
    scale_limit = 0.1,
    rotate_limit = 45,
    interpolation = 1,
    border_mode = 4,
    value = NULL,
    mask_value = NULL,
    shift_limit_x = NULL,
    shift_limit_y = NULL,
    always_apply = FALSE,
    p = 0.5
)
```

**Arguments**

<code>shift_limit</code>	<code>shift_limit</code>
<code>scale_limit</code>	<code>scale_limit</code>



rotate_limit	rotate_limit
interpolation	interpolation
border_mode	border_mode
value	value
mask_value	mask_value
shift_limit_x	shift_limit_x
shift_limit_y	shift_limit_y
always_apply	always_apply
p	p

**Value**

None  
None

**Targets**

image, mask, keypoints  
image, mask, keypoints

**Image types**

uint8, float32  
uint8, float32

---

icevision\_SingleSplitSplitter  
*SingleSplitSplitter*

---

**Description**

SingleSplitSplitter

**Usage**

icevision\_SingleSplitSplitter(...)

**Arguments**

... arguments to pass

**Value**

all items in a single group, without shuffling.

---

icevision\_SmallestMaxSize  
*SmallestMaxSize*

---

### Description

Rescale an image so that minimum side is equal to max\_size, keeping the aspect ratio of the initial image.

### Usage

```
icevision_SmallestMaxSize(  
    max_size = 1024,  
    interpolation = 1,  
    always_apply = FALSE,  
    p = 1  
)
```

### Arguments

max_size	max_size
interpolation	interpolation
always_apply	always_apply
p	p

### Value

None

### Targets

image, mask, bboxes, keypoints

### Image types

uint8, float32

---

icevision\_Solarize     *Solarize*

---

**Description**

Invert all pixel values above a threshold.

**Usage**

```
icevision_Solarize(threshold = 128, always_apply = FALSE, p = 0.5)
```

**Arguments**

threshold	threshold
always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

any

---

icevision\_ToFloat     *ToFloat*

---

**Description**

Divide pixel values by 'max\_value' to get a float32 output array where all values lie in the range [0, 1.0].

**Usage**

```
icevision_ToFloat(max_value = NULL, always_apply = FALSE, p = 1)
```

**Arguments**

max_value	max_value
always_apply	always_apply
p	p

**Details**

If 'max\_value' is NULL the transform will try to infer the maximum value by inspecting the data type of the input image. See Also: :class:`~albumentations.augmentations.transforms.FromFloat`

**Value**

None

**See Also**

:class:`~albumentations.augmentations.transforms.FromFloat`

**Targets**

image

**Image types**

any type

---

icevision_ToGray	<i>ToGray</i>
------------------	---------------

---

**Description**

Convert the input RGB image to grayscale. If the mean pixel value for the resulting image is greater than 127, invert the resulting grayscale image.

**Usage**

```
icevision_ToGray(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

icevision_ToSepia	<i>ToSepia</i>
-------------------	----------------

---

**Description**

Applies sepia filter to the input RGB image

**Usage**

```
icevision_ToSepia(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

**Targets**

image

**Image types**

uint8, float32

---

icevision_Transpose	<i>Transpose</i>
---------------------	------------------

---

**Description**

Transpose the input by swapping rows and columns.

**Usage**

```
icevision_Transpose(always_apply = FALSE, p = 0.5)
```

**Arguments**

always_apply	always_apply
p	p

**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

`icevision_VerticalFlip`*VerticalFlip*

---

**Description**

Flip the input vertically around the x-axis.

**Usage**`icevision_VerticalFlip(always_apply = FALSE, p = 0.5)`**Arguments**`always_apply`    `always_apply``p`                `p`**Value**

None

**Targets**

image, mask, bboxes, keypoints

**Image types**

uint8, float32

---

icnr_init	<i>Icnr_init</i>
-----------	------------------

---

**Description**

ICNR init of 'x', with 'scale' and 'init' function

**Usage**

```
icnr_init(x, scale = 2, init = nn()$init$kaiming_normal_)
```

**Arguments**

x	tensor
scale	int, scale
init	initializer

**Value**

None

---

IDMap	<i>IDMap</i>
-------	--------------

---

**Description**

Works like a dictionary that automatically assign values for new keys.

**Usage**

```
IDMap(initial_names = NULL)
```

**Arguments**

initial_names	initial_names
---------------	---------------

**Value**

None

Image

*Image*

---

**Description**

Image

**Usage**

Image(...)

**Arguments**

... parameters to pass

**Value**None

---

image2tensor

*Image2tensor*

---

**Description**

Transform image to byte tensor in 'c\*h\*w' dim order.

**Usage**

image2tensor(img)

**Arguments**

img image

**Value**

None



---

ImageBlock

*ImageBlock*

---

**Description**

A 'TransformBlock' for images of 'cls'

**Usage**

ImageBlock(...)

**Arguments**

... parameters to pass

**Value**

block

---

ImageBW\_create

*ImageBW\_create*

---

**Description**

Open an 'Image' from path 'fn'

**Usage**

ImageBW\_create(fn)

**Arguments**

fn file name

**Value**

None

---

 ImageDataLoaders\_from\_csv

*ImageDataLoaders from csv*


---

### Description

Create from 'path/csv\_fname' using 'fn\_col' and 'label\_col'

### Usage

```

ImageDataLoaders_from_csv(
  path,
  csv_fname = "labels.csv",
  header = "infer",
  delimiter = NULL,
  valid_pct = 0.2,
  seed = NULL,
  fn_col = 0,
  folder = NULL,
  suff = "",
  label_col = 1,
  label_delim = NULL,
  y_block = NULL,
  valid_col = NULL,
  item_tfms = NULL,
  batch_tfms = NULL,
  bs = 64,
  val_bs = NULL,
  size = NULL,
  shuffle_train = TRUE,
  device = NULL,
  ...
)

```

### Arguments

path	The folder where to work
csv_fname	csv file name
header	header
delimiter	delimiter
valid_pct	validation percentage
seed	random seed
fn_col	column name
folder	folder name

suff	suff
label_col	label column
label_delim	label delimiter
y_block	y_block
valid_col	validation column
item_tfms	One or several transforms applied to the items before batching them
batch_tfms	One or several transforms applied to the batches once they are formed
bs	batch size
val_bs	The batch size for the validation DataLoader (defaults to bs)
size	image size
shuffle_train	If we shuffle the training DataLoader or not
device	device name
...	additional parameters to pass

**Value**

None

---

ImageDataLoaders\_from\_dblock  
*ImageDataLoaders from dblock*

---

**Description**

Create a dataloaders from a given ‘dblock‘

**Usage**

```
ImageDataLoaders_from_dblock(
    dblock,
    source,
    path = ".",
    bs = 64,
    val_bs = NULL,
    shuffle_train = TRUE,
    device = NULL,
    ...
)
```

**Arguments**

dblock	dblock
source	source folder
path	The folder where to work
bs	batch size
val_bs	The batch size for the validation DataLoader (defaults to bs)
shuffle_train	If we shuffle the training DataLoader or not
device	device name
...	additional parameters to pass

**Value**

None

---

ImageDataLoaders\_from\_df

*ImageDataLoaders from df*


---

**Description**

Create from 'df' using 'fn\_col' and 'label\_col'

**Usage**

```
ImageDataLoaders_from_df(
    df,
    path = ".",
    valid_pct = 0.2,
    seed = NULL,
    fn_col = 0,
    folder = NULL,
    suff = "",
    label_col = 1,
    label_delim = NULL,
    y_block = NULL,
    valid_col = NULL,
    item_tfms = NULL,
    batch_tfms = NULL,
    bs = 64,
    val_bs = NULL,
    shuffle_train = TRUE,
    device = NULL,
    ...
)
```

**Arguments**

df	data frame
path	The folder where to work
valid_pct	validation percentage
seed	random seed
fn_col	column name
folder	folder name
suff	suff
label_col	label column
label_delim	label separator
y_block	y_block
valid_col	validation column
item_tfms	One or several transforms applied to the items before batching them
batch_tfms	One or several transforms applied to the batches once they are formed
bs	batch size
val_bs	The batch size for the validation DataLoader (defaults to bs)
shuffle_train	shuffle_train
device	device
...	additional parameters to pass

**Value**

None

---

ImageDataLoaders\_from\_folder  
*ImageDataLoaders from folder*

---

**Description**

Create from imagenet style dataset in 'path' with 'train' and 'valid' subfolders (or provide 'valid\_pct')

**Usage**

```
ImageDataLoaders_from_folder(
    path,
    train = "train",
    valid = "valid",
    valid_pct = NULL,
    seed = NULL,
    vocab = NULL,
```

```

    item_tfms = NULL,
    batch_tfms = NULL,
    bs = 64,
    val_bs = NULL,
    shuffle_train = TRUE,
    device = NULL,
    size = NULL,
    ...
)

```

### Arguments

path	The folder where to work
train	train data
valid	validation data
valid_pct	validation percentage
seed	random seed
vocab	vocabulary
item_tfms	One or several transforms applied to the items before batching them
batch_tfms	One or several transforms applied to the batches once they are formed
bs	batch size
val_bs	The batch size for the validation DataLoader (defaults to bs)
shuffle_train	If we shuffle the training DataLoader or not
device	device name
size	image size
...	additional parameters to pass

---

ImageDataLoaders\_from\_lists

*ImageDataLoaders from lists*

---

### Description

Create from list of 'fnames' and 'labels' in 'path'

### Usage

```

ImageDataLoaders_from_lists(
  path,
  fnames,
  labels,
  valid_pct = 0.2,
  seed = NULL,
)

```

```

    y_block = NULL,
    item_tfms = NULL,
    batch_tfms = NULL,
    bs = 64,
    val_bs = NULL,
    shuffle_train = TRUE,
    device = NULL,
    ...
)

```

### Arguments

path	The folder where to work
fnames	file names
labels	labels
valid_pct	validation percentage
seed	random seed
y_block	y_block
item_tfms	One or several transforms applied to the items before batching them
batch_tfms	One or several transforms applied to the batches once they are formed
bs	batch size
val_bs	The batch size for the validation DataLoader (defaults to bs)
shuffle_train	If we shuffle the training DataLoader or not
device	device name
...	additional parameters to pass

### Value

None

---

ImageDataLoaders\_from\_name\_re

*ImageDataLoaders from name regex*

---

### Description

Create from the name attrs of 'fnames' in 'path's with re expression 'pat'

**Usage**

```
ImageDataLoaders_from_name_re(
    path,
    fnames,
    pat,
    bs = 64,
    val_bs = NULL,
    shuffle_train = TRUE,
    device = NULL,
    item_tfms = NULL,
    batch_tfms = NULL,
    ...
)
```

**Arguments**

path	The folder where to work
fnames	folder names
pat	an argument that requires regex
bs	The batch size
val_bs	The batch size for the validation DataLoader (defaults to bs)
shuffle_train	If we shuffle the training DataLoader or not
device	device name
item_tfms	One or several transforms applied to the items before batching them
batch_tfms	One or several transforms applied to the batches once they are formed
...	additional parameters to pass

**Value**

None

**Examples**

```
## Not run:

URLs_PETS()

path = 'oxford-iiit-pet'

dls = ImageDataLoaders_from_name_re(
  path, fnames, pat='(.)_\\d+.jpg$',
  item_tfms = RandomResizedCrop(460, min_scale=0.75), bs = 10,
  batch_tfms = list(aug_transforms(size = 299, max_warp = 0),
                    Normalize_from_stats( imagenet_stats() )
  ),
  device = 'cuda'
```



```
)

## End(Not run)
```

---

```
ImageDataLoaders_from_path_func
    ImageDataLoaders from path function
```

---

### Description

Create from list of 'fnames' in 'path's with 'label\_func'

### Usage

```
ImageDataLoaders_from_path_func(
  path,
  fnames,
  label_func,
  valid_pct = 0.2,
  seed = NULL,
  item_tfms = NULL,
  batch_tfms = NULL,
  bs = 64,
  val_bs = NULL,
  shuffle_train = TRUE,
  device = NULL,
  ...
)
```

### Arguments

path	The folder where to work
fnames	file names
label_func	label function
valid_pct	The random percentage of the dataset to set aside for validation (with an optional seed)
seed	random seed
item_tfms	One or several transforms applied to the items before batching them
batch_tfms	One or several transforms applied to the batches once they are formed
bs	batch size
val_bs	The batch size for the validation DataLoader (defaults to bs)
shuffle_train	If we shuffle the training DataLoader or not
device	device name
...	additional parameters to pass

**Value**

None

---

 ImageDataLoaders\_from\_path\_re

*ImageDataLoaders from path re*


---

**Description**

Create from list of 'fnames' in 'path's with re expression 'pat'

**Usage**

```

ImageDataLoaders_from_path_re(
    path,
    fnames,
    pat,
    valid_pct = 0.2,
    seed = NULL,
    item_tfms = NULL,
    batch_tfms = NULL,
    bs = 64,
    val_bs = NULL,
    shuffle_train = TRUE,
    device = NULL,
    ...
)

```

**Arguments**

path	The folder where to work
fnames	file names
pat	an argument that requires regex
valid_pct	The random percentage of the dataset to set aside for validation (with an optional seed)
seed	random seed
item_tfms	One or several transforms applied to the items before batching them
batch_tfms	One or several transforms applied to the batches once they are formed
bs	batch size
val_bs	The batch size for the validation DataLoader (defaults to bs)
shuffle_train	If we shuffle the training DataLoader or not
device	device name
...	additional parameters to pass

**Value**

None

---

imagenet_stats	<i>Imagenet statistics</i>
----------------	----------------------------

---

**Description**

Imagenet statistics

**Usage**

imagenet\_stats()

**Value**

vector

**Examples**

```
## Not run:  
  
imagenet_stats()  
  
## End(Not run)
```

---

Image_create	<i>Image_create</i>
--------------	---------------------

---

**Description**

Open an 'Image' from path 'fn'

**Usage**

Image\_create(fn)

**Arguments**

fn                   file name

**Value**

None

---

Image_open	<i>Image_open</i>
------------	-------------------

---

**Description**

Opens and identifies the given image file.

**Usage**

```
Image_open(fp, mode = "r")
```

**Arguments**

fp	fp
mode	mode

**Value**

None

---

Image_resize	<i>Resize</i>
--------------	---------------

---

**Description**

Returns a resized copy of this image.

**Usage**

```
Image_resize(img, size, resample = 3, box = NULL, reducing_gap = NULL)
```

**Arguments**

img	image
size	size
resample	resample
box	box
reducing_gap	reducing_gap

**Value**

None

---

InceptionModule	<i>InceptionModule</i>
-----------------	------------------------

---

**Description**

The inception Module from 'ni' inputs to len('kss')\*'nb\_filters'+ 'bottleneck\_size'

**Usage**

```
InceptionModule(  
    ni,  
    nb_filters = 32,  
    kss = c(39, 19, 9),  
    bottleneck_size = 32,  
    stride = 1  
)
```

**Arguments**

ni	number of input channels
nb_filters	the number of filters
kss	kernel size
bottleneck_size	bottleneck size
stride	stride

**Value**

module

---

IndexSplitter	<i>Index Splitter</i>
---------------	-----------------------

---

**Description**

Split 'items' so that 'val\_idx' are in the validation set and the others in the training set

**Usage**

```
IndexSplitter(valid_idx)
```

**Arguments**

valid_idx	The indices to use for the validation set (defaults to a random split otherwise)
-----------	--

**Value**

None

---

`init`

---

*Wandb init*

---

**Description**

Initialize a wandb Run.

**Usage**`init(...)`**Arguments**`...` parameters to pass**Value**

wandb Run object

None

**see https**`//docs.wandb.com/library/init`

---

`init_default`

---

*Init\_default*

---

**Description**

Initialize 'm' weights with 'func' and set 'bias' to 0.

**Usage**`init_default(m, func = nn()$init$kaiming_normal_)`**Arguments**`m` parameters`func` function**Value**

None

---

init_linear	<i>Init_linear</i>
-------------	--------------------

---

**Description**

Init\_linear

**Usage**

```
init_linear(m, act_func = NULL, init = "auto", bias_std = 0.01)
```

**Arguments**

m	parameter
act_func	activation function
init	initializer
bias_std	bias standard deviation

**Value**

None

---

install_fastai	<i>Install fastai</i>
----------------	-----------------------

---

**Description**

Install fastai

**Usage**

```
install_fastai(
  version,
  gpu = FALSE,
  cuda_version = "10",
  overwrite = FALSE,
  extra_pkgs = c("timm", "fastinference[interp]"),
  TPU = FALSE
)
```

**Arguments**

version	specify version
gpu	installation of gpu
cuda_version	if gpu true, then cuda version is required. By default it is 10
overwrite	will install all the dependencies
extra_pkgs	character vector of additional packages
TPU	official way to install Pytorch-XLA 1.7

**Value**

None

---

InstanceNorm	<i>InstanceNorm</i>
--------------	---------------------

---

**Description**

InstanceNorm layer with 'nf' features and 'ndim' initialized depending on 'norm\_type'.

**Usage**

```
InstanceNorm(
  nf,
  ndim = 2,
  norm_type = 5,
  affine = TRUE,
  eps = 1e-05,
  momentum = 0.1,
  track_running_stats = FALSE
)
```

**Arguments**

nf	input shape
ndim	dimension number
norm_type	normalization type
affine	affine
eps	epsilon
momentum	momentum
track_running_stats	track running statistics

**Value**

None



---

IntToFloatTensor	<i>IntToFloatTensor</i>
------------------	-------------------------

---

**Description**

Transform image to float tensor, optionally dividing by 255 (e.g. for images).

**Usage**

```
IntToFloatTensor(div = 255, div_mask = 1)
```

**Arguments**

div	divide value
div_mask	divide mask

**Value**

None

---

InvisibleTensor	<i>Invisible Tensor</i>
-----------------	-------------------------

---

**Description**

Invisible Tensor

**Usage**

```
InvisibleTensor(x)
```

**Arguments**

x	tensor
---	--------

**Value**

None

---

in_channels	<i>In_channels</i>
-------------	--------------------

---

**Description**

Return the shape of the first weight layer in 'm'.

**Usage**

```
in_channels(m)
```

**Arguments**

m                    parameters

**Value**

None

---

is_rmarkdown	<i>Is Rmarkdown?</i>
--------------	----------------------

---

**Description**

Is Rmarkdown?

**Usage**

```
is_rmarkdown()
```

**Value**

logical True/False

---

Jaccard	<i>Jaccard</i>
---------	----------------

---

**Description**

Jaccard score for single-label classification problems

**Usage**

```
Jaccard(
  axis = -1,
  labels = NULL,
  pos_label = 1,
  average = "binary",
  sample_weight = NULL
)
```

**Arguments**

axis	axis
labels	labels
pos_label	pos_label
average	average
sample_weight	sample_weight

**Value**

None

---

JaccardCoeff	<i>JaccardCoeff</i>
--------------	---------------------

---

**Description**

Implementation of the Jaccard coefficient that is lighter in RAM

**Usage**

```
JaccardCoeff(axis = 1)
```

**Arguments**

axis	axis
------	------

**Value**

None

---

JaccardMulti	<i>JaccardMulti</i>
--------------	---------------------

---

**Description**

Jaccard score for multi-label classification problems

**Usage**

```
JaccardMulti(
  thresh = 0.5,
  sigmoid = TRUE,
  labels = NULL,
  pos_label = 1,
  average = "macro",
  sample_weight = NULL
)
```

**Arguments**

thresh	thresh
sigmoid	sigmoid
labels	labels
pos_label	pos_label
average	average
sample_weight	sample_weight

**Value**

None

---

kg	<i>Kaggle module</i>
----	----------------------

---

**Description**

Kaggle module

**Usage**

```
kg()
```

**Value**

None

---

L

*L*

---

### Description

Behaves like a list of ‘items’ but can also index with list of indices or masks

### Usage

L(...)

### Arguments

... arguments to pass

---

L1LossFlat

*L1LossFlat*

---

### Description

Flattens input and output, same as nn\$L1LossFlat

### Usage

L1LossFlat(...)

### Arguments

... parameters to pass

### Value

Loss object

---

l2_reg	<i>L2_reg</i>
--------	---------------

---

**Description**

L2 regularization as adding 'wd\*p' to 'p\$grad'

**Usage**

```
l2_reg(p, lr, wd, do_wd = TRUE, ...)
```

**Arguments**

p	p
lr	learning rate
wd	weight decay
do_wd	do_wd
...	additional arguments to pass

**Value**

None

**Examples**

```
## Not run:

tst_param = function(val, grad = NULL) {
  "Create a tensor with `val` and a gradient of `grad` for testing"
  res = tensor(val) %>% float()

  if(is.null(grad)) {
    grad = tensor(val / 10)
  } else {
    grad = tensor(grad)
  }

  res$grad = grad %>% float()
  res
}
p = tst_param(1., 0.1)
l2_reg(p, 1., 0.1)

## End(Not run)
```

---

LabeledBBox

*LabeledBBox*


---

**Description**

Basic type for a list of bounding boxes in an image

**Usage**

LabeledBBox(...)

**Arguments**

... parameters to pass

**Value**

None

---

LabelSmoothingCrossEntropy

*LabelSmoothingCrossEntropy*


---

**Description**

Same as 'nn\$Module', but no need for subclasses to call 'super().\_\_init\_\_'

**Usage**

LabelSmoothingCrossEntropy(eps = 0.1, reduction = "mean")

**Arguments**

eps                   epsilon  
reduction           reduction, defaults to mean

**Value**

Loss object

---

LabelSmoothingCrossEntropyFlat

*LabelSmoothingCrossEntropyFlat*

---

**Description**

Same as 'nn\$Module', but no need for subclasses to call 'super().\_\_init\_\_'

**Usage**

LabelSmoothingCrossEntropyFlat(...)

**Arguments**

... parameters to pass

**Value**

Loss object

---

Lamb

*Lamb*

---

**Description**

Lamb

**Usage**

Lamb(...)

**Arguments**

... parameters to pass

**Value**

None



---

Lambda	<i>Lambda</i>
--------	---------------

---

**Description**

An easy way to create a pytorch layer for a simple ‘func’

**Usage**

```
Lambda(func)
```

**Arguments**

func	function
------	----------

**Value**

None

---

lamb_step	<i>Lamb_step</i>
-----------	------------------

---

**Description**

Step for LAMB with ‘lr’ on ‘p’

**Usage**

```
lamb_step(p, lr, mom, step, sqr_mom, grad_avg, sqr_avg, eps, ...)
```

**Arguments**

p	p
lr	learning rate
mom	momentum
step	step
sqr_mom	sqr momentum
grad_avg	gradient average
sqr_avg	sqr average
eps	epsilon
...	additional arguments to pass

**Value**

None

---

`language_model_learner`*Language\_model\_learner*

---

### Description

Create a ‘Learner’ with a language model from ‘dls’ and ‘arch’.

### Usage

```
language_model_learner(  
  dls,  
  arch,  
  config = NULL,  
  drop_mult = 1,  
  backwards = FALSE,  
  pretrained = TRUE,  
  pretrained_fnames = NULL,  
  opt_func = Adam(),  
  lr = 0.001,  
  cbs = NULL,  
  metrics = NULL,  
  path = NULL,  
  model_dir = "models",  
  wd = NULL,  
  wd_bn_bias = FALSE,  
  train_bn = TRUE,  
  moms = list(0.95, 0.85, 0.95),  
  ...  
)
```

### Arguments

<code>dls</code>	<code>dls</code>
<code>arch</code>	<code>arch</code>
<code>config</code>	<code>config</code>
<code>drop_mult</code>	<code>drop_mult</code>
<code>backwards</code>	<code>backwards</code>
<code>pretrained</code>	<code>pretrained</code>
<code>pretrained_fnames</code>	<code>pretrained_fnames</code>
<code>opt_func</code>	<code>opt_func</code>
<code>lr</code>	<code>lr</code>
<code>cbs</code>	<code>cbs</code>

metrics	metrics
path	path
model_dir	model_dir
wd	wd
wd_bn_bias	wd_bn_bias
train_bn	train_bn
moms	moms
...	additional arguments

**Value**

None

---

Larc

*Larc*


---

**Description**

Larc

**Usage**

Larc(...)

**Arguments**

... parameters to pass

**Value**

None

---

larc_layer_lr	<i>Larc_layer_lr</i>
---------------	----------------------

---

**Description**

Computes the local lr before weight decay is applied

**Usage**

```
larc_layer_lr(p, lr, trust_coeff, wd, eps, clip = TRUE, ...)
```

**Arguments**

p	p
lr	learning rate
trust_coeff	trust_coeff
wd	weight decay
eps	epsilon
clip	clip
...	additional arguments to pass

**Value**

None

---

larc_step	<i>Larc_step</i>
-----------	------------------

---

**Description**

Step for LARC 'local\_lr' on 'p'

**Usage**

```
larc_step(p, local_lr, grad_avg = NULL, ...)
```

**Arguments**

p	p
local_lr	local learning rate
grad_avg	gradient average
...	additional args to pass

**Value**

None

---

layer\_info

*Layer\_info*

---

**Description**

Return layer infos of 'model' on 'xb' (only support batch first inputs)

**Usage**

```
layer_info(learn, ...)
```

**Arguments**

learn	learner/model
...	additional arguments

**Value**

None

---

Learner

*Learner*

---

**Description**

Learner

**Usage**

```
Learner(...)
```

**Arguments**

...	parameters to pass
-----	--------------------

**Value**

None

**Examples**

```

## Not run:

model = LitModel()

data = Data_Loaders(model$train_data_loader(), model$val_data_loader())$cuda()

learn = Learner(data, model, loss_func = F$cross_entropy, opt_func = Adam,
                metrics = accuracy)

## End(Not run)

```

---

length	<i>Length</i>
--------	---------------

---

**Description**

Length

**Usage**

```

## S3 method for class 'torch.Tensor'
length(x)

```

**Arguments**

x            tensor

**Value**

tensor

---

length.fastai.torch_core.TensorMask	<i>Length</i>
-------------------------------------	---------------

---

**Description**

Length

**Usage**

```

## S3 method for class 'fastai.torch_core.TensorMask'
length(x)

```

**Arguments**

x                    tensor

**Value**

tensor

---

less                    *Less*

---

**Description**

Less

**Usage**

```
## S3 method for class 'torch.Tensor'
a < b
```

**Arguments**

a                    tensor  
b                    tensor

**Value**

tensor

---

less\_or\_equal            *Less or equal*

---

**Description**

Less or equal

**Usage**

```
## S3 method for class 'torch.Tensor'
a <= b
```

**Arguments**

a                    tensor  
b                    tensor

**Value**

tensor

---

LightingTfm	<i>LightingTfm</i>
-------------	--------------------

---

**Description**

Apply 'fs' to the logits

**Usage**

```
LightingTfm(fs, ...)
```

**Arguments**

fs	fs
...	parameters to pass

**Value**

None

---

LinBnDrop	<i>LinBnDrop</i>
-----------	------------------

---

**Description**

Module grouping 'BatchNorm1d', 'Dropout' and 'Linear' layers

**Usage**

```
LinBnDrop(n_in, n_out, bn = TRUE, p = 0, act = NULL, lin_first = FALSE)
```

**Arguments**

n_in	input shape
n_out	output shape
bn	bn
p	probability
act	activation
lin_first	linear first

**Value**

None



---

LinearDecoder	<i>LinearDecoder</i>
---------------	----------------------

---

**Description**

To go on top of a RNNCore module and create a Language Model.

**Usage**

```
LinearDecoder(n_out, n_hid, output_p = 0.1, tie_encoder = NULL, bias = TRUE)
```

**Arguments**

n_out	n_out
n_hid	n_hid
output_p	output_p
tie_encoder	tie_encoder
bias	bias

**Value**

None

---

LitModel	<i>Lit Model</i>
----------	------------------

---

**Description**

Lit Model

**Usage**

```
LitModel()
```

**Value**

model

LMDataLoader

*LMDataLoader*

---

**Description**

A 'DataLoader' suitable for language modeling

**Usage**

```
LMDataLoader(  
    dataset,  
    lens = NULL,  
    cache = 2,  
    bs = 64,  
    seq_len = 72,  
    num_workers = 0,  
    shuffle = FALSE,  
    verbose = FALSE,  
    do_setup = TRUE,  
    pin_memory = FALSE,  
    timeout = 0L,  
    batch_size = NULL,  
    drop_last = FALSE,  
    indexed = NULL,  
    n = NULL,  
    device = NULL  
)
```

**Arguments**

dataset	dataset
lens	lens
cache	cache
bs	bs
seq_len	seq_len
num_workers	num_workers
shuffle	shuffle
verbose	verbose
do_setup	do_setup
pin_memory	pin_memory
timeout	timeout
batch_size	batch_size
drop_last	drop_last

indexed	indexed
n	n
device	device

**Value**

text loader

---

LMLearner

*LMLearner*


---

**Description**

Add functionality to ‘TextLearner‘ when dealing with a language model

Add functionality to ‘TextLearner‘ when dealing with a language model

**Usage**

```
LMLearner(
  dls,
  model,
  alpha = 2,
  beta = 1,
  moms = list(0.8, 0.7, 0.8),
  loss_func = NULL,
  opt_func = Adam(),
  lr = 0.001,
  splitter = trainable_params(),
  cbs = NULL,
  metrics = NULL,
  path = NULL,
  model_dir = "models",
  wd = NULL,
  wd_bn_bias = FALSE,
  train_bn = TRUE
)
```

```
LMLearner(
  dls,
  model,
  alpha = 2,
  beta = 1,
  moms = list(0.8, 0.7, 0.8),
  loss_func = NULL,
  opt_func = Adam(),
  lr = 0.001,
```

```

    splitter = trainable_params(),
    cbs = NULL,
    metrics = NULL,
    path = NULL,
    model_dir = "models",
    wd = NULL,
    wd_bn_bias = FALSE,
    train_bn = TRUE
)

```

### Arguments

dls	dls
model	model
alpha	alpha
beta	beta
moms	moms
loss_func	loss_func
opt_func	opt_func
lr	lr
splitter	splitter
cbs	cbs
metrics	metrics
path	path
model_dir	model_dir
wd	wd
wd_bn_bias	wd_bn_bias
train_bn	train_bn

### Value

text loader  
None

---

LMLearner\_predict      *LMLearner\_predict*

---

### Description

Return 'text' and the 'n\_words' that come after

**Usage**

```
MLearner_predict(  
    text,  
    n_words = 1,  
    no_unk = TRUE,  
    temperature = 1,  
    min_p = NULL,  
    no_bar = FALSE,  
    decoder = decode_spec_tokens(),  
    only_last_word = FALSE  
)
```

**Arguments**

text	text
n_words	n_words
no_unk	no_unk
temperature	temperature
min_p	min_p
no_bar	no_bar
decoder	decoder
only_last_word	only_last_word

**Value**

None

---

loaders

*Loaders*

---

**Description**

a loader from Catalyst

**Usage**

```
loaders()
```

**Value**

None

**Examples**

```
## Not run:

# trigger download
loaders()

## End(Not run)
```

---

load_dataset	<i>Load_dataset</i>
--------------	---------------------

---

**Description**

A helper function for getting a DataLoader for images in the folder ‘test\_path’, with batch size ‘bs’, and number of workers ‘num\_workers’

**Usage**

```
load_dataset(test_path, bs = 4, num_workers = 4)
```

**Arguments**

test_path	test path (directory)
bs	batch size
num_workers	number of workers

**Value**

None

---

load_ignore_keys	<i>Load_ignore_keys</i>
------------------	-------------------------

---

**Description**

Load ‘wgt’s in ‘model’ ignoring the names of the keys, just taking parameters in order

**Usage**

```
load_ignore_keys(model, wgt's)
```

**Arguments**

model	model
wgts	wgts

**Value**

None

---

load_image	<i>Load_image</i>
------------	-------------------

---

**Description**

Open and load a 'PIL.Image' and convert to 'mode'

**Usage**

```
load_image(fn, mode = NULL)
```

**Arguments**

fn	file name
mode	mode

**Value**

None

---

load_learner	<i>Load_learner</i>
--------------	---------------------

---

**Description**

Load a 'Learner' object in 'fname', optionally putting it on the 'cpu'

**Usage**

```
load_learner(fname, cpu = TRUE)
```

**Arguments**

fname	fname
cpu	cpu or not

**Value**

learner object

---

load_model_text	<i>Load_model_text</i>
-----------------	------------------------

---

**Description**

Load 'model' from 'file' along with 'opt' (if available, and if 'with\_opt')

**Usage**

```
load_model_text(
  file,
  model,
  opt,
  with_opt = NULL,
  device = NULL,
  strict = TRUE
)
```

**Arguments**

file	file
model	model
opt	opt
with_opt	with_opt
device	device
strict	strict

**Value**

None

---

load_pre_models	<i>Timm models</i>
-----------------	--------------------

---

**Description**

Timm models

**Usage**

```
load_pre_models()
```

**Value**

None



---

load_tokenized_csv	<i>Load_tokenized_csv</i>
--------------------	---------------------------

---

**Description**

Utility function to quickly load a tokenized csv and the corresponding counter

**Usage**

```
load_tokenized_csv(fname)
```

**Arguments**

fname	file name
-------	-----------

**Value**

None

---

log	<i>Log</i>
-----	------------

---

**Description**

Log

**Usage**

```
## S3 method for class 'torch.Tensor'  
log(x, base = exp(1))
```

**Arguments**

x	tensor
base	base parameter

**Value**

tensor

log.fastai.torch\_core.TensorMask

*Log*

---

### Description

Log

### Usage

```
## S3 method for class 'fastai.torch_core.TensorMask'  
log(x, base = exp(1))
```

### Arguments

x	tensor
base	base parameter

### Value

tensor

---

log1p

*Log1p*

---

### Description

Log1p

### Usage

```
## S3 method for class 'torch.Tensor'  
log1p(x)
```

### Arguments

x	tensor
---	--------

### Value

tensor

---

log1p.fastai.torch\_core.TensorMask  
*Log1p*

---

**Description**

Log1p

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
log1p(x)
```

**Arguments**

x            tensor

**Value**

tensor

---

logical\_and            *Logical\_and*

---

**Description**

Logical\_and

**Usage**

```
## S3 method for class 'torch.Tensor'  
x & y
```

**Arguments**

x            tensor  
y            tensor

**Value**

tensor

---

logical_not_	<i>Logical_not</i>
--------------	--------------------

---

**Description**

Logical\_not

**Usage**

```
## S3 method for class 'torch.Tensor'  
!x
```

**Arguments**

x	tensor
---	--------

**Value**

tensor

---

logical_or	<i>Logical_or</i>
------------	-------------------

---

**Description**

Logical\_or

**Usage**

```
## S3 method for class 'torch.Tensor'  
x | y
```

**Arguments**

x	tensor
y	tensor

**Value**

tensor

---

login	<i>Wandb login</i>
-------	--------------------

---

**Description**

Log in to W&B.

**Usage**

```
login(anonymous = NULL, key = NULL, relogin = NULL, host = NULL, force = NULL)
```

**Arguments**

anonymous	must,never,allow,false,true
key	API key (secret)
relogin	relogin or not
host	host address
force	whether to force a user to be logged into wandb when running a script

**Value**

None

---

Lookahead	<i>Lookahead</i>
-----------	------------------

---

**Description**

Lookahead

**Usage**

```
Lookahead(...)
```

**Arguments**

...	parameters to pass
-----	--------------------

**Value**

None

---

LossMetric	<i>LossMetric</i>
------------	-------------------

---

**Description**

Create a metric from 'loss\_func.attr' named 'nm'

**Usage**

```
LossMetric(attr, nm = NULL)
```

**Arguments**

attr	attr
nm	nm

**Value**

None

---

lr_find	<i>Lr_find</i>
---------	----------------

---

**Description**

Launch a mock training to find a good learning rate, return lr\_min, lr\_steep if 'suggestions' is TRUE

**Usage**

```
lr_find(
  object,
  start_lr = 1e-07,
  end_lr = 10,
  num_it = 100,
  stop_div = TRUE,
  ...
)
```

**Arguments**

object	learner
start_lr	starting learning rate
end_lr	end learning rate
num_it	number of iterations
stop_div	stop div or not
...	additional arguments to pass

**Value**

data frame

**Examples**

```
## Not run:

model %>% lr_find()
model %>% plot_lr_find(dpi = 200)

## End(Not run)
```

---

mae	<i>MAE</i>
-----	------------

---

**Description**

Mean absolute error between ‘inp’ and ‘targ’.

**Usage**

```
mae(inp, targ)
```

**Arguments**

inp	predictions
targ	targets

**Value**

None

---

make_vocab	<i>Make_vocab</i>
------------	-------------------

---

**Description**

Create a vocab of ‘max\_vocab’ size from ‘Counter’ ‘count’ with items present more than ‘min\_freq’

**Usage**

```
make_vocab(count, min_freq = 3, max_vocab = 60000, special_toks = NULL)
```

**Arguments**

count	count
min_freq	min_freq
max_vocab	max_vocab
special_toks	special_toks

**Value**

None

---

mask2bbox	<i>Mask2bbox</i>
-----------	------------------

---

**Description**

Mask2bbox

**Usage**

mask2bbox(mask, convert = TRUE)

**Arguments**

mask	mask
convert	to R matrix

**Value**

tensor

---

MaskBlock	<i>MaskBlock</i>
-----------	------------------

---

**Description**

A ‘TransformBlock’ for segmentation masks, potentially with ‘codes’

**Usage**

MaskBlock(codes = NULL)

**Arguments**

codes	codes
-------	-------

**Value**

block



---

masked_concat_pool	<i>Masked_concat_pool</i>
--------------------	---------------------------

---

**Description**

Pool 'MultiBatchEncoder' outputs into one vector [last\_hidden, max\_pool, avg\_pool]

**Usage**

```
masked_concat_pool(output, mask, bptt)
```

**Arguments**

output	output
mask	mask
bptt	bptt

**Value**

None

---

MaskFreq	<i>Mask Freq</i>
----------	------------------

---

**Description**

Google SpecAugment frequency masking from <https://arxiv.org/abs/1904.08779>.

**Usage**

```
MaskFreq(num_masks = 1, size = 20, start = NULL, val = NULL)
```

**Arguments**

num_masks	number of masks
size	size
start	starting point
val	value

**Value**

None

---

MaskTime	<i>MaskTime</i>
----------	-----------------

---

**Description**

Google SpecAugment time masking from <https://arxiv.org/abs/1904.08779>.

**Usage**

```
MaskTime(num_masks = 1, size = 20, start = NULL, val = NULL)
```

**Arguments**

num_masks	number of masks
size	size
start	starting point
val	value

**Value**

None

---

Mask_create	<i>Mask_create</i>
-------------	--------------------

---

**Description**

Delegates (`'__call__'`, `'decode'`, `'setup'`) to (`'encodes'`, `'decodes'`, `'setups'`) if `'split_idx'` matches

**Usage**

```
Mask_create(enc = NULL, dec = NULL, split_idx = NULL, order = NULL)
```

**Arguments**

enc	encoder
dec	decoder
split_idx	split by index
order	order

**Value**

None

---

mask_from_blur	<i>Mask from blur</i>
----------------	-----------------------

---

**Description**

Mask from blur

**Usage**

```
mask_from_blur(img, window, sigma = 0.3, thresh = 0.05, remove_max = TRUE)
```

**Arguments**

img	image
window	windowing effect
sigma	sigma
thresh	threshold point
remove_max	remove maximum or not

---

mask_rcnn_infer_dl	<i>Mask RCNN infer dataloader</i>
--------------------	-----------------------------------

---

**Description**

A ‘DataLoader’ with a custom ‘collate\_fn’ that batches items as required for inferring the model.

**Usage**

```
mask_rcnn_infer_dl(dataset, batch_tfms = NULL, ...)
```

**Arguments**

dataset	Possibly a ‘Dataset’ object, but more generally, any ‘Sequence’ that returns records.
batch_tfms	Transforms to be applied at the batch level. <b>**dataloader_kwargs</b> : Keyword arguments that will be internally passed to a Pytorch ‘DataLoader’. The parameter ‘collate_fn’ is already defined internally and cannot be passed here.
...	additional arguments

**Value**

None

---

mask_rcnn_learner	<i>MaskRCNN learner</i>
-------------------	-------------------------

---

**Description**

Fastai ‘Learner’ adapted for MaskRCNN.

**Usage**

```
mask_rcnn_learner(dls, model, cbs = NULL, ...)
```

**Arguments**

dls	‘Sequence’ of ‘DataLoaders’ passed to the ‘Learner’. The first one will be used for training and the second for validation.
model	The model to train.
cbs	Optional ‘Sequence’ of callbacks.
...	learner_kwargs: Keyword arguments that will be internally passed to ‘Learner’.

**Value**

model

---

mask_rcnn_model	<i>MaskRCNN model</i>
-----------------	-----------------------

---

**Description**

MaskRCNN model implemented by torchvision.

**Usage**

```
mask_rcnn_model(  
    num_classes,  
    backbone = NULL,  
    remove_internal_transforms = TRUE,  
    pretrained = TRUE  
)
```

**Arguments**

num_classes	Number of classes.
backbone	Backbone model to use. Defaults to a resnet50_fpn model.
remove_internal_transforms	The torchvision model internally applies transforms like resizing and normalization, but we already do this at the 'Dataset' level, so it's safe to remove those internal transforms.
pretrained	Argument passed to 'maskrcnn_resnet50_fpn' if 'backbone is NULL'. By default it is set to TRUE: this is generally used when training a new model (transfer learning). 'pretrained = FALSE' is used during inference (prediction) for cases where the users have their own pretrained weights. <b>**mask_rcnn_kwargs</b> : Keyword arguments that internally are going to be passed to 'torchvision.models.detection.mask_rcnn.MaskRCNN'.

**Value**

model

---

mask\_rcnn\_predict\_dl *Mask RCNN predict dataloader*

---

**Description**

Mask RCNN predict dataloader

**Usage**

```
mask_rcnn_predict_dl(model, infer_dl, show_pbar = TRUE)
```

**Arguments**

model	model
infer_dl	infer_dl
show_pbar	show_pbar

**Value**

None

---

mask\_rcnn\_train\_dl      *MaskRCNN train dataloader*

---

**Description**

A ‘DataLoader’ with a custom ‘collate\_fn’ that batches items as required for training the model.

**Usage**

```
mask_rcnn_train_dl(dataset, batch_tfms = NULL, ...)
```

**Arguments**

dataset	Possibly a ‘Dataset’ object, but more generally, any ‘Sequence’ that returns records.
batch_tfms	Transforms to be applied at the batch level.
...	dataloader_kwargs: Keyword arguments that will be internally passed to a PyTorch ‘DataLoader’. The parameter ‘collate_fn’ is already defined internally and cannot be passed here.

**Value**

None

---

mask\_rcnn\_valid\_dl      *MaskRSNN valid dataloader*

---

**Description**

A ‘DataLoader’ with a custom ‘collate\_fn’ that batches items as required for training the model.

**Usage**

```
mask_rcnn_valid_dl(dataset, batch_tfms = NULL, ...)
```

**Arguments**

dataset	Possibly a ‘Dataset’ object, but more generally, any ‘Sequence’ that returns records.
batch_tfms	Transforms to be applied at the batch level.
...	dataloader_kwargs: Keyword arguments that will be internally passed to a PyTorch ‘DataLoader’. The parameter ‘collate_fn’ is already defined internally and cannot be passed here.

**Value**

None

---

mask_tensor	<i>Mask_tensor</i>
-------------	--------------------

---

**Description**

Mask elements of 'x' with 'neutral' with probability '1-p'

**Usage**

```
mask_tensor(x, p = 0.5, neutral = 0, batch = FALSE)
```

**Arguments**

x	tensor
p	probability
neutral	neutral
batch	batch

**Value**

None

---

match_embeds	<i>Match_embeds</i>
--------------	---------------------

---

**Description**

Convert the embedding in 'old\_wgts' to go from 'old\_vocab' to 'new\_vocab'.

**Usage**

```
match_embeds(old_wgts, old_vocab, new_vocab)
```

**Arguments**

old_wgts	old_wgts
old_vocab	old_vocab
new_vocab	new_vocab

**Value**

None

---

MatthewsCorrCoef	<i>MatthewsCorrCoef</i>
------------------	-------------------------

---

**Description**

Matthews correlation coefficient for single-label classification problems

**Usage**

```
MatthewsCorrCoef(...)
```

**Arguments**

... parameters to pass

**Value**

None

---

MatthewsCorrCoefMulti	<i>MatthewsCorrCoefMulti</i>
-----------------------	------------------------------

---

**Description**

Matthews correlation coefficient for multi-label classification problems

**Usage**

```
MatthewsCorrCoefMulti(thresh = 0.5, sigmoid = TRUE, sample_weight = NULL)
```

**Arguments**

thresh	thresh
sigmoid	sigmoid
sample_weight	sample_weight

**Value**

None



---

max	<i>Max</i>
-----	------------

---

**Description**

Max

**Usage**

```
## S3 method for class 'torch.Tensor'
max(a, ..., na.rm = FALSE)
```

**Arguments**

a	tensor
...	additional parameters
na.rm	remove NAs

**Value**

tensor

---

max.fastai.torch_core.TensorMask	<i>Max</i>
----------------------------------	------------

---

**Description**

Max

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'
max(a, ..., na.rm = FALSE)
```

**Arguments**

a	tensor
...	additional parameters
na.rm	remove NAs

**Value**

tensor

---

MaxPool	<i>MaxPool</i>
---------	----------------

---

**Description**

nn.MaxPool layer for 'ndim'

**Usage**

```
MaxPool(ks = 2, stride = NULL, padding = 0, ndim = 2, ceil_mode = FALSE)
```

**Arguments**

ks	kernel size
stride	the stride of the window. Default value is kernel_size
padding	implicit zero padding to be added on both sides
ndim	dimension number
ceil_mode	when True, will use ceil instead of floor to compute the output shape

**Value**

None

---

maybe_unsqueeze	<i>Maybe_unsqueeze</i>
-----------------	------------------------

---

**Description**

Add empty dimension if it is a rank 1 tensor/array

**Usage**

```
maybe_unsqueeze(x)
```

**Arguments**

x	R array/matrix/tensor
---	-----------------------

**Value**

array

---

MCDropoutCallback      *MCDropoutCallback*


---

**Description**

Turns on dropout during inference, allowing you to call `Learner$get_preds` multiple times to approximate your model uncertainty using Monte Carlo Dropout. <https://arxiv.org/pdf/1506.02142.pdf>

**Usage**

```
MCDropoutCallback(...)
```

**Arguments**

```
...                    arguments to pass
```

**Value**

```
None
```

---

```
mean.fastai.torch_core.TensorMask
```

*Mean of tensor*

---

**Description**

Mean of tensor

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'
mean(x, ...)
```

**Arguments**

```
x                    tensor
...                    additional parameters to pass
```

**Value**

```
tensor
```

mean.torch.Tensor      *Mean of tensor*

---

**Description**

Mean of tensor

**Usage**

```
## S3 method for class 'torch.Tensor'  
mean(x, ...)
```

**Arguments**

x                    tensor  
...                  additional parameters to pass

**Value**

tensor

---

medical                *Medical module*

---

**Description**

Medical module

**Usage**

```
medical()
```

**Value**

None

---

MergeLayer	<i>MergeLayer</i>
------------	-------------------

---

**Description**

Merge a shortcut with the result of the module by adding them or concatenating them if 'dense=TRUE'.

**Usage**

```
MergeLayer(dense = FALSE)
```

**Arguments**

dense	dense
-------	-------

**Value**

None

---

metrics	<i>Metrics module</i>
---------	-----------------------

---

**Description**

Metrics module

**Usage**

```
metrics()
```

**Value**

None

---

migrating_ignite	<i>Ignite module</i>
------------------	----------------------

---

**Description**

Ignite module

**Usage**

```
migrating_ignite()
```

**Value**

None

migrating\_lightning     *Lightning module*

---

**Description**

Lightning module

**Usage**

migrating\_lightning()

**Value**

None

---

migrating\_pytorch     *Pytorch module*

---

**Description**

Pytorch module

**Usage**

migrating\_pytorch()

**Value**

None

---

min     *Min*

---

**Description**

Min

**Usage**

```
## S3 method for class 'torch.Tensor'  
min(a, ..., na.rm = FALSE)
```

**Arguments**

a	tensor
...	additional parameters
na.rm	remove NAs

**Value**

tensor

---

*min.fastai.torch\_core.TensorMask*  
*Min*

---

**Description**

Min

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
min(a, ..., na.rm = FALSE)
```

**Arguments**

a	tensor
...	additional parameters
na.rm	remove NAs

**Value**

tensor

---

mish *Mish*

---

**Description**

Mish

**Usage**

mish(x)

**Arguments**

x                    tensor

**Value**

None

---

MishJitAutoFn

*MishJitAutoFn*

---

**Description**

Records operation history and defines formulas for differentiating ops.

**Usage**

MishJitAutoFn(...)

**Arguments**

...                    parameters to pass

**Value**

None

---

Mish\_

*Class Mish*

---

**Description**

Class Mish

**Usage**

Mish\_(...)

**Arguments**

...                    parameters to pass

**Value**

None



---

MixHandler

*MixHandler*

---

**Description**

A handler class for implementing ‘MixUp’ style scheduling

**Usage**

```
MixHandler(alpha = 0.5)
```

**Arguments**

alpha            alpha

**Value**

None

---

MixUp

*MixUp*

---

**Description**

Implementation of <https://arxiv.org/abs/1710.09412>

**Usage**

```
MixUp(alpha = 0.4)
```

**Arguments**

alpha            alpha

**Value**

None

---

`ModelResetter`*ModelResetter*

---

**Description**

Callback that resets the model at each validation/training step

**Usage**

```
ModelResetter(...)
```

**Arguments**

... arguments to pass

**Value**

None

---

`model_sizes`*Model\_sizes*

---

**Description**

Pass a dummy input through the model 'm' to get the various sizes of activations.

**Usage**

```
model_sizes(m, size = list(64, 64))
```

**Arguments**

m m parameter

size size

**Value**

None

---

Module	<i>Module module</i>
--------	----------------------

---

**Description**

Module module

**Usage**

Module()

**Value**

None

---

Module_test	<i>NN module</i>
-------------	------------------

---

**Description**

NN module

**Usage**

Module\_test()

**Value**

None

---

momentum_step	<i>Momentum_step</i>
---------------	----------------------

---

**Description**

Step for SGD with momentum with 'lr'

**Usage**

momentum\_step(p, lr, grad\_avg, ...)

**Arguments**

p	p
lr	learning rate
grad_avg	grad average
...	additional arguments to pass

**Value**

None

---

most_confused	<i>Most_confused</i>
---------------	----------------------

---

**Description**

Sorted descending list of largest non-diagonal entries of confusion matrix, presented as actual, predicted, number of occurrences.

**Usage**

```
most_confused(interp, min_val = 1)
```

**Arguments**

interp	interpretation object
min_val	minimum value

**Value**

data frame

---

mse	<i>MSE</i>
-----	------------

---

**Description**

Mean squared error between 'inp' and 'targ'.

**Usage**

```
mse(inp, targ)
```

**Arguments**

inp	predictions
targ	targets

**Value**

None

**Examples**

```
## Not run:  
  
model = dls %>% tabular_learner(layers=c(200,100,100,200),  
metrics = list(mse(),rmse() )  
  
## End(Not run)
```

---

MSELossFlat

*MSELossFlat*

---

**Description**

Flattens input and output, same as nn\$MSELoss

**Usage**

```
MSELossFlat(...)
```

**Arguments**

... parameters to pass

**Value**

Loss object

---

msle	<i>MSLE</i>
------	-------------

---

**Description**

Mean squared logarithmic error between 'inp' and 'targ'.

**Usage**

```
msle(inp, targ)
```

**Arguments**

inp	predictions
targ	targets

**Value**

None

---

MultiCategorize	<i>MultiCategorize</i>
-----------------	------------------------

---

**Description**

Reversible transform of multi-category strings to 'vocab' id

**Usage**

```
MultiCategorize(vocab = NULL, add_na = FALSE)
```

**Arguments**

vocab	vocabulary
add_na	add NA

**Value**

None

---

MultiCategoryBlock     *MultiCategoryBlock*

---

**Description**

‘TransformBlock’ for multi-label categorical targets

**Usage**

```
MultiCategoryBlock(encoded = FALSE, vocab = NULL, add_na = FALSE)
```

**Arguments**

encoded	encoded or not
vocab	vocabulary
add_na	add NA

**Value**

Block object

---

```
multiplygit add -A && git commit -m 'staging all files'
```

*Multiply*

---

**Description**

Multiply

**Usage**

```
## S3 method for class 'torch.Tensor'
a * b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

MultiTargetLoss	<i>MultiTargetLoss</i>
-----------------	------------------------

---

**Description**

Provides the ability to apply different loss functions to multi-modal targets/predictions

**Usage**

```
MultiTargetLoss(...)
```

**Arguments**

... additional arguments

**Value**

None

---

narrow	<i>Modify tensor</i>
--------	----------------------

---

**Description**

Modify tensor

**Usage**

```
narrow(tensor, slice)
```

**Arguments**

tensor torch tensor

slice dimension

**Value**

tensor



---

Net

*Net*

---

**Description**

Net model from Migrating\_Pytorch

**Usage**

Net()

**Value**

model

**Examples**

```
## Not run:  
  
Net()  
  
## End(Not run)
```

---

nn

*NN module*

---

**Description**

NN module

**Usage**

nn()

**Value**

None

---

nn_loss	<i>Fastai custom loss</i>
---------	---------------------------

---

**Description**

Fastai custom loss

**Usage**

```
nn_loss(loss_fn, name = "Custom_Loss")
```

**Arguments**

loss_fn	pass custom model function
name	set name for nn_module

**Value**

None

---

nn_module	<i>Fastai NN module</i>
-----------	-------------------------

---

**Description**

Fastai NN module

**Usage**

```
nn_module(model_fn, name = "Custom_Model", gpu = TRUE)
```

**Arguments**

model_fn	pass custom model function
name	set name for nn_module
gpu	move model to GPU

**Value**

None

---

NoiseColor	<i>NoiseColor module</i>
------------	--------------------------

---

**Description**

NoiseColor module

**Usage**

NoiseColor()

**Value**

None

---

NoneReduce	<i>NoneReduce</i>
------------	-------------------

---

**Description**

A context manager to evaluate 'loss\_func' with none reduce.

**Usage**

NoneReduce(loss\_func)

**Arguments**

loss\_func      loss function

**Value**

None

---

noop

*Noop*

---

**Description**

Noop

**Usage**

noop(...)

**Arguments**

... parameters to pass

**Value**

None

---

Normalize

*Normalize*

---

**Description**

Normalize the continuous variables.

**Usage**

Normalize(cat\_names, cont\_names)

**Arguments**

cat\_names cat\_names

cont\_names cont\_names

**Value**

None

---

NormalizeTS	<i>NormalizeTS</i>
-------------	--------------------

---

**Description**

Normalize the x variables.

**Usage**

```
NormalizeTS(enc = NULL, dec = NULL, split_idx = NULL, order = NULL)
```

**Arguments**

enc	encoder
dec	decoder
split_idx	split by index
order	order

**Value**

None

---

Normalize_from_stats	<i>Normalize from stats</i>
----------------------	-----------------------------

---

**Description**

Normalize from stats

**Usage**

```
Normalize_from_stats(mean, std, dim = 1, ndim = 4, cuda = TRUE)
```

**Arguments**

mean	mean
std	standard deviation
dim	dimension
ndim	number of dimensions
cuda	cuda or not

**Value**

list

---

norm_apply_denorm	<i>Norm_apply_denorm</i>
-------------------	--------------------------

---

**Description**

Normalize 'x' with 'nrm', then apply 'f', then denormalize

**Usage**

```
norm_apply_denorm(x, f, nrm)
```

**Arguments**

x	tensor
f	function
nrm	nrm

**Value**

None

---

not_equal_to	<i>Not equal</i>
--------------	------------------

---

**Description**

Not equal

**Usage**

```
## S3 method for class 'torch.Tensor'
a != b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

not_equal_to_mask_	<i>Not equal</i>
--------------------	------------------

---

**Description**

Not equal

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
a != b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

not__mask	<i>Logical_not</i>
-----------	--------------------

---

**Description**

Logical\_not

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
!x
```

**Arguments**

x	tensor
---	--------

**Value**

tensor

---

Numericalize	<i>Numericalize</i>
--------------	---------------------

---

**Description**

Reversible transform of tokenized texts to numericalized ids

**Usage**

```
Numericalize(
  vocab = NULL,
  min_freq = 3,
  max_vocab = 60000,
  special_toks = NULL,
  pad_tok = NULL
)
```

**Arguments**

vocab	vocab
min_freq	min_freq
max_vocab	max_vocab
special_toks	special_toks
pad_tok	pad_tok

**Value**

None

---

num_features_model	<i>Num_features_model</i>
--------------------	---------------------------

---

**Description**

Return the number of output features for 'm'.

**Usage**

```
num_features_model(m)
```

**Arguments**

m	m parameter
---	-------------

**Value**

None



---

n\_px

*N\_px*


---

**Description**

int(x=0) -> integer

**Usage**

n\_px(img)

**Arguments**

img            image

**Value**

None

---

OldRandomCrop

*OldRandomCrop*


---

**Description**

Randomly crop an image to 'size'

**Usage**

OldRandomCrop(size, pad\_mode = "zeros", ...)

**Arguments**

size            size  
pad\_mode        padding mode  
...              additional arguments

**Value**

None

---

one_batch	<i>One batch</i>
-----------	------------------

---

**Description**

One batch

**Usage**

```
one_batch(object, convert = FALSE, ...)
```

**Arguments**

object	data loader
convert	to R matrix
...	additional parameters to pass

**Value**

tensor

**Examples**

```
## Not run:

# get batch from data loader
batch = dls %>% one_batch()

## End(Not run)
```

---

OpenAudio	<i>OpenAudio</i>
-----------	------------------

---

**Description**

Transform that creates AudioTensors from a list of files.

**Usage**

```
OpenAudio(items)
```

**Arguments**

items	vector, items
-------	---------------

**Value**

None

---

Optimizer

*Optimizer*

---

**Description**

Optimizer

**Usage**

Optimizer(...)

**Arguments**

... parameters to pass

**Value**

None

---

OptimWrapper

*OptimWrapper*

---

**Description**

OptimWrapper

**Usage**

OptimWrapper(...)

**Arguments**

... parameters to pass

**Value**

None

---

optim_metric	<i>Optim metric</i>
--------------	---------------------

---

**Description**

Replace metric 'f' with a version that optimizes argument 'argname'

**Usage**

```
optim_metric(f, argname, bounds, tol = 0.01, do_neg = TRUE, get_x = FALSE)
```

**Arguments**

f	f
argname	argname
bounds	bounds
tol	tol
do_neg	do_neg
get_x	get_x

**Value**

None

---

or_mask	<i>Logical_or</i>
---------	-------------------

---

**Description**

Logical\_or

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'
x | y
```

**Arguments**

x	tensor
y	tensor

**Value**

tensor

---

os	<i>Operating system</i>
----	-------------------------

---

**Description**

Operating system

**Usage**

os()

**Value**

vector

---

os_environ_tpu	<i>An environment supporting TPUs</i>
----------------	---------------------------------------

---

**Description**

An environment supporting TPUs

**Usage**

```
os_environ_tpu(text = "COLAB_TPU_ADDR")
```

**Arguments**

text            string to pass to environment

**Value**

None

---

pad\_conv\_norm\_relu      *Pad\_conv\_norm\_relu*

---

### Description

Pad\_conv\_norm\_relu

### Usage

```
pad_conv_norm_relu(  
  ch_in,  
  ch_out,  
  pad_mode,  
  norm_layer,  
  ks = 3,  
  bias = TRUE,  
  pad = 1,  
  stride = 1,  
  activ = TRUE,  
  init = nn()$init$kaiming_normal_,  
  init_gain = 0.02  
)
```

### Arguments

ch_in	input
ch_out	output
pad_mode	padding mode
norm_layer	normalization layer
ks	kernel size
bias	bias
pad	padding
stride	stride
activ	activation
init	initializer
init_gain	init gain

### Value

None

---

pad_input	<i>Pad_input</i>
-----------	------------------

---

**Description**

Function that collect 'samples' and adds padding

**Usage**

```
pad_input(
    samples,
    pad_idx = 1,
    pad_fields = 0,
    pad_first = FALSE,
    backwards = FALSE
)
```

**Arguments**

samples	samples
pad_idx	pad_idx
pad_fields	pad_fields
pad_first	pad_first
backwards	backwards

**Value**

None

---

pad_input_chunk	<i>Pad_input_chunk</i>
-----------------	------------------------

---

**Description**

Pad 'samples' by adding padding by chunks of size 'seq\_len'

**Usage**

```
pad_input_chunk(samples, pad_idx = 1, pad_first = TRUE, seq_len = 72)
```

**Arguments**

samples	samples
pad_idx	pad_idx
pad_first	pad_first
seq_len	seq_len

**Value**

None

---

parallel	<i>Parallel</i>
----------	-----------------

---

**Description**

Applies ‘func’ in parallel to ‘items’, using ‘n\_workers’

**Usage**

```
parallel(f, items, ...)
```

**Arguments**

f	file names
items	items
...	additional arguments

**Value**

None

---

parallel_tokenize	<i>Parallel_tokenize</i>
-------------------	--------------------------

---

**Description**

Calls optional ‘setup’ on ‘tok’ before launching ‘TokenizeWithRules’ using ‘parallel\_gen

**Usage**

```
parallel_tokenize(items, tok = NULL, rules = NULL, n_workers = 6)
```

**Arguments**

items	items
tok	tokenizer
rules	rules
n_workers	n_workers

**Value**

None



---

params

*Params*

---

**Description**

Return all parameters of 'm'

**Usage**

params(m)

**Arguments**

m                    parameters

**Value**

None

---

ParamScheduler

*ParamScheduler*

---

**Description**

Schedule hyper-parameters according to 'scheds'

**Usage**

ParamScheduler(scheds)

**Arguments**

scheds                scheds

**Value**

None

parent\_label      *Parent\_label*

---

**Description**

Label 'item' with the parent folder name.

**Usage**

```
parent_label(o)
```

**Arguments**

o                    string, dir path

**Value**

vector

---

parsers\_AreasMixin      *AreasMixin*

---

**Description**

Adds areas method to parser

**Usage**

```
parsers_AreasMixin(...)
```

**Arguments**

...                    arguments to pass

**Value**

None

---

`parsers_BBoxesMixin`    *BBoxesMixin*

---

**Description**

Adds bboxes method to parser

**Usage**

`parsers_BBoxesMixin(...)`

**Arguments**

...            arguments to pass

**Value**

None

---

`parsers_FasterRCNN`    *Faster RCNN*

---

**Description**

Parser with required mixins for Faster RCNN.

**Usage**

`parsers_FasterRCNN(...)`

**Arguments**

...            arguments to pass

**Value**

None

parsers\_FilepathMixin *FilepathMixin*

---

**Description**

Adds filepath method to parser

**Usage**

```
parsers_FilepathMixin(...)
```

**Arguments**

... arguments to pass

**Value**

None

---

parsers\_ImageidMixin *Imageid Mixin*

---

**Description**

Adds imageid method to parser

**Usage**

```
parsers_ImageidMixin(...)
```

**Arguments**

... arguments to pass

**Value**

None

---

*parsers\_IsCrowdsMixin* *IsCrowdsMixin*

---

**Description**

Adds iscrowds method to parser

**Usage**

`parsers_IsCrowdsMixin(...)`

**Arguments**

... arguments to pass

**Value**

None

---

*parsers\_LabelsMixin* *LabelsMixin*

---

**Description**

Adds labels method to parser

**Usage**

`parsers_LabelsMixin(...)`

**Arguments**

... arguments to pass

**Value**

None

`parsers_MaskRCNN`      *Mask RCNN*

---

**Description**

Parser with required mixins for Mask RCNN.

**Usage**

```
parsers_MaskRCNN(...)
```

**Arguments**

...                    arguments to pass

**Value**

None

---

`parsers_MasksMixin`      *MasksMixin*

---

**Description**

Adds masks method to parser

**Usage**

```
parsers_MasksMixin(...)
```

**Arguments**

...                    arguments to pass

**Value**

None

*parsers\_SizeMixin*      *SizeMixin*

**Description**

Adds image\_width\_height method to parser

**Usage**

`parsers_SizeMixin(...)`

**Arguments**

...                    arguments to pass

**Value**

None

*parsers\_voc*              *Voc parser*

**Description**

Voc parser

**Usage**

`parsers_voc(annotations_dir, images_dir, class_map, masks_dir = NULL)`

**Arguments**

annotations\_dir            annotations\_dir  
 images\_dir                images\_dir  
 class\_map                 class\_map  
 masks\_dir                 masks\_dir

**Value**

None

---

partial	<i>Partial</i>
---------	----------------

---

**Description**

partial(func, \*args, \*\*keywords) - new function with partial application

**Usage**

```
partial(...)
```

**Arguments**

```
... additional arguments
```

**Value**

None

**Examples**

```
## Not run:

generator = basic_generator(out_size = 64, n_channels = 3, n_extra_layers = 1)
critic     = basic_critic(in_size = 64, n_channels = 3, n_extra_layers = 1,
                          act_cls = partial(nn$LeakyReLU, negative_slope = 0.2))

## End(Not run)
```

---

PartialDL	<i>PartialDL</i>
-----------	------------------

---

**Description**

Select randomly partial quantity of data at each epoch

**Usage**

```
PartialDL(
  dataset = NULL,
  bs = NULL,
  partial_n = NULL,
  shuffle = FALSE,
  num_workers = NULL,
```



```

    verbose = FALSE,
    do_setup = TRUE,
    pin_memory = FALSE,
    timeout = 0,
    batch_size = NULL,
    drop_last = FALSE,
    indexed = NULL,
    n = NULL,
    device = NULL,
    persistent_workers = FALSE
  )

```

### Arguments

dataset	dataset
bs	bs
partial_n	partial_n
shuffle	shuffle
num_workers	num_workers
verbose	verbose
do_setup	do_setup
pin_memory	pin_memory
timeout	timeout
batch_size	batch_size
drop_last	drop_last
indexed	indexed
n	n
device	device
persistent_workers	persistent_workers

### Value

None

---

 PartialLambda

*Partial Lambda*


---

### Description

Layer that applies ‘partial(func, ...)’

**Usage**

```
PartialLambda(func)
```

**Arguments**

```
func          function
```

**Value**

```
None
```

---

```
pca
```

```
PCA
```

---

**Description**

Compute PCA of 'x' with 'k' dimensions.

**Usage**

```
pca(object, k = 3, convert = TRUE)
```

**Arguments**

```
object        an object to apply PCA
k             number of dimensions
convert       to R matrix
```

**Value**

```
tensor
```

---

```
PearsonCorrCoef
```

```
PearsonCorrCoef
```

---

**Description**

Pearson correlation coefficient for regression problem

**Usage**

```
PearsonCorrCoef(  
  dim_argmax = NULL,  
  activation = "no",  
  thresh = NULL,  
  to_np = FALSE,  
  invert_arg = FALSE,  
  flatten = TRUE  
)
```

**Arguments**

dim_argmax	dim_argmax
activation	activation
thresh	thresh
to_np	to_np
invert_arg	invert_arg
flatten	flatten

**Value**

None

---

Perplexity

*Perplexity*

---

**Description**

Perplexity

**Usage**

```
Perplexity(...)
```

**Arguments**

... parameters to pass

**Value**

None

---

Pipeline	<i>Pipeline</i>
----------	-----------------

---

**Description**

A pipeline of composed (for encode/decode) transforms, setup with types

**Usage**

```
Pipeline(funcs = NULL, split_idx = NULL)
```

**Arguments**

funcs	functions
split_idx	split by index

**Value**

None

---

PixelShuffle_ICNR	<i>PixelShuffle_ICNR</i>
-------------------	--------------------------

---

**Description**

Upsample by 'scale' from 'ni' filters to 'nf' (default 'ni'), using 'nn.PixelShuffle'.

**Usage**

```
PixelShuffle_ICNR(
  ni,
  nf = NULL,
  scale = 2,
  blur = FALSE,
  norm_type = 3,
  act_cls = nn()$ReLU
)
```

**Arguments**

ni	input shape
nf	number of features / outputs
scale	scale
blur	blur
norm_type	normalziation type
act_cls	activation

**Value**

None

---

plot	<i>Plot dicom</i>
------	-------------------

---

**Description**

Plot dicom

**Usage**`plot(x, y, ..., dpi = 100)`**Arguments**

x	model
y	y axis
...	parameters to pass
dpi	dots per inch

**Value**

None

---

plot_bs_find	<i>Plot_bs_find</i>
--------------	---------------------

---

**Description**

Plot\_bs\_find

**Usage**`plot_bs_find(object, ..., dpi = 250)`**Arguments**

object	model
...	additional arguments
dpi	dots per inch

**Value**

None

---

plot\_confusion\_matrix *Plot\_confusion\_matrix*

---

### Description

Plot the confusion matrix, with 'title' and using 'cmap'.

### Usage

```
plot_confusion_matrix(  
  interp,  
  normalize = FALSE,  
  title = "Confusion matrix",  
  cmap = "Blues",  
  norm_dec = 2,  
  plot_txt = TRUE,  
  figsize = c(4, 4),  
  ...,  
  dpi = 120  
)
```

### Arguments

interp	interpretation object
normalize	normalize
title	title
cmap	color map
norm_dec	norm dec
plot_txt	plot text
figsize	plot size
...	additional parameters to pass
dpi	dots per inch

### Value

None

### Examples

```
## Not run:  
  
interp = ClassificationInterpretation_from_learner(model)  
interp %>% plot_confusion_matrix(dpi = 90, figsize = c(6,6))
```

```
## End(Not run)
```

---

plot_loss	<i>Plot_loss</i>
-----------	------------------

---

### Description

Plot the losses from ‘skip\_start’ and onward

### Usage

```
plot_loss(object, skip_start = 5, with_valid = TRUE, dpi = 200)
```

### Arguments

object	model
skip_start	n points to skip the start
with_valid	with validation
dpi	dots per inch

### Value

None

---

plot_lr_find	<i>Plot_lr_find</i>
--------------	---------------------

---

### Description

Plot the result of an LR Finder test (won’t work if you didn’t do ‘lr\_find(learn)’ before)

### Usage

```
plot_lr_find(object, skip_end = 5, dpi = 250)
```

### Arguments

object	model
skip_end	n points to skip the end
dpi	dots per inch

### Value

None

---

plot_top_losses	<i>Plot_top_losses</i>
-----------------	------------------------

---

## Description

Plot\_top\_losses

## Usage

```
plot_top_losses(interp, k, largest = TRUE, figsize = c(7, 5), ..., dpi = 90)
```

## Arguments

interp	interpretation object
k	number of images
largest	largest
figsize	plot size
...	additional parameters to pass
dpi	dots per inch

## Value

None

## Examples

```
## Not run:  
  
# get interperetation from learn object, the model.  
interp = ClassificationInterpretation_from_learner(learn)  
interp %>% plot_top_losses(k = 9, figsize = c(15,11))  
  
## End(Not run)
```



---

PointBlock

*PointBlock*

---

**Description**

A 'TransformBlock' for points in an image

**Usage**

```
PointBlock()
```

**Value**

None

---

PointScaler

*PointScaler*

---

**Description**

Scale a tensor representing points

**Usage**

```
PointScaler(do_scale = TRUE, y_first = FALSE)
```

**Arguments**

do\_scale      do scale

y\_first      y first

**Value**

None

---

PooledSelfAttention2d *PooledSelfAttention2d*

---

**Description**

Pooled self attention layer for 2d.

**Usage**

PooledSelfAttention2d(n\_channels)

**Arguments**

n\_channels      number of channels

**Value**

None

---

PoolFlatten      *PoolFlatten*

---

**Description**

Combine ‘nn.AdaptiveAvgPool2d‘ and ‘Flatten‘.

**Usage**

PoolFlatten(pool\_type = "Avg")

**Arguments**

pool\_type      pooling type

**Value**

None

---

PoolingLinearClassifier  
*PoolingLinearClassifier*

---

**Description**

Create a linear classifier with pooling

**Usage**

```
PoolingLinearClassifier(dims, ps, bptt, y_range = NULL)
```

**Arguments**

dims	dims
ps	ps
bptt	bptt
y_range	y_range

**Value**

None

---

pow	<i>Pow</i>
-----	------------

---

**Description**

Pow

**Usage**

```
## S3 method for class 'torch.Tensor'
a ^ b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

Precision	<i>Precision</i>
-----------	------------------

---

**Description**

Precision for single-label classification problems

**Usage**

```
Precision(
  axis = -1,
  labels = NULL,
  pos_label = 1,
  average = "binary",
  sample_weight = NULL
)
```

**Arguments**

axis	axis
labels	labels
pos_label	pos_label
average	average
sample_weight	sample_weight

**Value**

None

---

PrecisionMulti	<i>PrecisionMulti</i>
----------------	-----------------------

---

**Description**

Precision for multi-label classification problems

**Usage**

```
PrecisionMulti(
  thresh = 0.5,
  sigmoid = TRUE,
  labels = NULL,
  pos_label = 1,
  average = "macro",
  sample_weight = NULL
)
```

**Arguments**

thresh	thresh
sigmoid	sigmoid
labels	labels
pos_label	pos_label
average	average
sample_weight	sample_weight

**Value**

None

---

`predict.fastai.learner.Learner`  
*Predict*

---

**Description**

Prediction on 'item', fully decoded, loss function decoded and probabilities

**Usage**

```
## S3 method for class 'fastai.learner.Learner'  
predict(object, row, ...)
```

**Arguments**

object	the model
row	row
...	additional arguments to pass

**Value**

data frame

---

```
predict.fastai.tabular.learner.TabularLearner
    Predict
```

---

**Description**

Prediction on 'item', fully decoded, loss function decoded and probabilities

**Usage**

```
## S3 method for class 'fastai.tabular.learner.TabularLearner'
predict(object, row, ...)
```

**Arguments**

object	the model
row	row
...	additional arguments to pass

**Value**

data frame

---

```
preplexity    Perplexity
```

---

**Description**

Perplexity (exponential of cross-entropy loss) for Language Models

**Usage**

```
preplexity(...)
```

**Arguments**

...	parameters to pass
-----	--------------------

**Value**

None

---

PreprocessAudio	<i>Preprocess Audio</i>
-----------------	-------------------------

---

**Description**

Creates an audio tensor and run the basic preprocessing transforms on it.

**Usage**

```
PreprocessAudio(sample_rate = 16000, force_mono = TRUE, crop_signal_to = NULL)
```

**Arguments**

sample_rate	sample rate
force_mono	force mono or not
crop_signal_to	int, crop signal

**Details**

Used while preprocessing the audios, this is not a 'Transform'.

**Value**

None

---

preprocess_audio_folder	<i>Preprocess audio folder</i>
-------------------------	--------------------------------

---

**Description**

Preprocess audio files in 'path' in parallel using 'n\_workers'

**Usage**

```
preprocess_audio_folder(  
  path,  
  folders = NULL,  
  output_dir = NULL,  
  sample_rate = 16000,  
  force_mono = TRUE,  
  crop_signal_to = NULL  
)
```

**Arguments**

path	directory, path
folders	folders
output_dir	output directory
sample_rate	sample rate
force_mono	force mono or not
crop_signal_to	int, crop signal

**Value**

None

---

pre_process_squad	<i>Pre_process_squad</i>
-------------------	--------------------------

---

**Description**

Pre\_process\_squad

**Usage**

```
pre_process_squad(row, hf_arch, hf_tokenizer)
```

**Arguments**

row	row in dataframe
hf_arch	architecture
hf_tokenizer	tokenizer

**Value**

None



---

```
print.fastai.learner.Learner  
    Print model
```

---

**Description**

Print model

**Usage**

```
## S3 method for class 'fastai.learner.Learner'  
print(x, ...)
```

**Arguments**

x	object
...	additional parameters to pass

**Value**

None

---

```
print.fastai.tabular.learner.TabularLearner  
    Print tabular model
```

---

**Description**

Print tabular model

**Usage**

```
## S3 method for class 'fastai.tabular.learner.TabularLearner'  
print(x, ...)
```

**Arguments**

x	model
...	additional parameters to pass

**Value**

None

---

```
print.pydicom.dataset.FileDataset
```

*Dicom*

---

**Description**

prints dicom file

**Usage**

```
## S3 method for class 'pydicom.dataset.FileDataset'  
print(x, ...)
```

**Arguments**

x	dicom file
...	additional parameters to pass

**Value**

None

---

```
python_path
```

*Python path*

---

**Description**

Python path

**Usage**

```
python_path()
```

**Value**

None

---

py\_apply

*Py\_apply*

---

**Description**

Pandas apply

**Usage**

py\_apply(df, ...)

**Arguments**

df                    dataframe  
...                    additional arguments

**Value**

dataframe

---

QHAdam

*QHAdam*

---

**Description**

QHAdam

**Usage**

QHAdam(...)

**Arguments**

...                    parameters to pass

**Value**

None

---

qhadam_step	<i>Qhadam_step</i>
-------------	--------------------

---

**Description**

Qhadam\_step

**Usage**

qhadam\_step(p, lr, mom, sqr\_mom, sqr\_avg, nu\_1, nu\_2, step, grad\_avg, eps, ...)

**Arguments**

p	p
lr	learning rate
mom	momentum
sqr_mom	sqr momentum
sqr_avg	sqr average
nu_1	nu_1
nu_2	nu_2
step	step
grad_avg	gradient average
eps	epsilon
...	additional arguments to pass

**Value**

None

---

QRNN	<i>QRNN</i>
------	-------------

---

**Description**

Apply a multiple layer Quasi-Recurrent Neural Network (QRNN) to an input sequence.

**Usage**

```
QRNN(  
    input_size,  
    hidden_size,  
    n_layers = 1,  
    batch_first = TRUE,  
    dropout = 0,  
    bidirectional = FALSE,  
    save_prev_x = FALSE,  
    zoneout = 0,  
    window = NULL,  
    output_gate = TRUE  
)
```

**Arguments**

input_size	input_size
hidden_size	hidden_size
n_layers	n_layers
batch_first	batch_first
dropout	dropout
bidirectional	bidirectional
save_prev_x	save_prev_x
zoneout	zoneout
window	window
output_gate	output_gate

**Value**

None

---

QRNNLayer

*QRNNLayer*

---

**Description**

Apply a single layer Quasi-Recurrent Neural Network (QRNN) to an input sequence.

**Usage**

```
QRNNLayer(
    input_size,
    hidden_size = NULL,
    save_prev_x = FALSE,
    zoneout = 0,
    window = 1,
    output_gate = TRUE,
    batch_first = TRUE,
    backward = FALSE
)
```

**Arguments**

input_size	input_size
hidden_size	hidden_size
save_prev_x	save_prev_x
zoneout	zoneout
window	window
output_gate	output_gate
batch_first	batch_first
backward	backward

**Value**

None

---

R2Score

*R2Score*

---

**Description**

R2 score between predictions and targets

**Usage**

```
R2Score(sample_weight = NULL)
```

**Arguments**

sample_weight	sample_weight
---------------	---------------

**Value**

None

---

RAdam	<i>RAdam</i>
-------	--------------

---

**Description**

RAdam

**Usage**

RAdam(...)

**Arguments**

... parameters to pass

**Value**

None

---

radam_step	<i>Radam_step</i>
------------	-------------------

---

**Description**

Step for RAdam with 'lr' on 'p'

**Usage**

radam\_step(p, lr, mom, step, sqr\_mom, grad\_avg, sqr\_avg, eps, beta, ...)

**Arguments**

p	p
lr	learning rate
mom	momentum
step	step
sqr_mom	sqr momentum
grad_avg	grad average
sqr_avg	sqr average
eps	epsilon
beta	beta
...	additional arguments to pass

**Value**

None

---

 RandomCrop

*RandomCrop*


---

**Description**

Randomly crop an image to 'size'

**Usage**

RandomCrop(size, ...)

**Arguments**

size	size
...	additional arguments

**Value**

None

---

RandomErasing

*RandomErasing*


---

**Description**

Randomly selects a rectangle region in an image and randomizes its pixels.

**Usage**

RandomErasing(p = 0.5, sl = 0, sh = 0.3, min\_aspect = 0.3, max\_count = 1)

**Arguments**

p	probability
sl	sl
sh	sh
min_aspect	minimum aspect
max_count	maximum count

**Value**

None



---

RandomResizedCrop      *RandomResizedCrop*

---

**Description**

Picks a random scaled crop of an image and resize it to 'size'

**Usage**

```
RandomResizedCrop(
    size,
    min_scale = 0.08,
    ratio = list(0.75, 1.3333333333333333),
    resamples = list(2, 0),
    val_xtra = 0.14
)
```

**Arguments**

size	size
min_scale	minimum scale
ratio	ratio
resamples	resamples
val_xtra	validation xtra

**Value**

None

---

RandomResizedCropGPU      *RandomResizedCropGPU*

---

**Description**

Picks a random scaled crop of an image and resize it to 'size'

**Usage**

```
RandomResizedCropGPU(
    size,
    min_scale = 0.08,
    ratio = list(0.75, 1.3333333333333333),
    mode = "bilinear",
    valid_scale = 1
)
```

**Arguments**

size	size
min_scale	minimum scale
ratio	ratio
mode	mode
valid_scale	validation scale

**Value**

None

---

RandomSplitter	<i>RandomSplitter</i>
----------------	-----------------------

---

**Description**

Create function that splits ‘items’ between train/val with ‘valid\_pct’ randomly.

**Usage**

```
RandomSplitter(valid_pct = 0.2, seed = NULL)
```

**Arguments**

valid_pct	validation percentatge split
seed	random seed

**Value**

None

---

RandPair	<i>RandPair</i>
----------	-----------------

---

**Description**

a random image from domain B, resulting in a random pair of images from domain A and B.

**Usage**

```
RandPair(itemsB)
```

**Arguments**

itemsB	a random image from domain B
--------	------------------------------

**Value**

None

---

RandTransform	<i>RandTransform</i>
---------------	----------------------

---

**Description**

A transform that before\_call its state at each ‘\_\_call\_\_’

**Usage**

```
RandTransform(p = 1, nm = NULL, before_call = NULL, ...)
```

**Arguments**

p	probability
nm	nm
before_call	before call
...	additional arguments to pass

**Value**

None

---

ranger	<i>Ranger</i>
--------	---------------

---

**Description**

Convenience method for ‘Lookahead’ with ‘RAdam’

**Usage**

```
ranger(
  p,
  lr,
  mom = 0.95,
  wd = 0.01,
  eps = 1e-06,
  sqr_mom = 0.99,
  beta = 0,
  decouple_wd = TRUE
)
```

**Arguments**

p	p
lr	learning rate
mom	momentum
wd	weight decay
eps	epsilon
sqr_mom	sqr momentum
beta	beta
decouple_wd	decouple weight decay

**Value**

None

---

RatioResize

*RatioResize*

---

**Description**

Resizes the biggest dimension of an image to 'max\_sz' maintaining the aspect ratio

**Usage**

```
RatioResize(max_sz, resamples = list(2, 0), ...)
```

**Arguments**

max_sz	maximum sz
resamples	resamples
...	additional arguments

**Value**

None

---

ReadTSBatch	<i>ReadTSBatch</i>
-------------	--------------------

---

**Description**

A transform that always take lists as items

**Usage**

```
ReadTSBatch(to)
```

**Arguments**

to	output from TSDDataTable function
----	-----------------------------------

**Value**

None

---

Recall	<i>Recall</i>
--------	---------------

---

**Description**

Recall for single-label classification problems

**Usage**

```
Recall(
  axis = -1,
  labels = NULL,
  pos_label = 1,
  average = "binary",
  sample_weight = NULL
)
```

**Arguments**

axis	axis
labels	labels
pos_label	pos_label
average	average
sample_weight	sample_weight

**Value**

None

---

 RecallMulti

*RecallMulti*


---

**Description**

Recall for multi-label classification problems

**Usage**

```
RecallMulti(
  thresh = 0.5,
  sigmoid = TRUE,
  labels = NULL,
  pos_label = 1,
  average = "macro",
  sample_weight = NULL
)
```

**Arguments**

thresh	thresh
sigmoid	sigmoid
labels	labels
pos_label	pos_label
average	average
sample_weight	sample_weight

**Value**

None

---

ReduceLROnPlateau

*ReduceLROnPlateau*


---

**Description**

ReduceLROnPlateau

**Usage**

```
ReduceLROnPlateau(...)
```

**Arguments**

...	parameters to pass
-----	--------------------

**Value**

None

**Examples**

```
## Not run:

URLs_MNIST_SAMPLE()
# transformations
tfms = aug_transforms(do_flip = FALSE)
path = 'mnist_sample'
bs = 20

#load into memory
data = ImageDataLoaders_from_folder(path, batch_tfms = tfms, size = 26, bs = bs)

learn = cnn_learner(data, resnet18(), metrics = accuracy, path = getwd())

learn %>% fit_one_cycle(10, 1e-2, cbs = ReduceLROnPlateau(monitor='valid_loss', patience = 1))

## End(Not run)
```

---

RegressionBlock

*RegressionBlock*

---

**Description**

‘TransformBlock‘ for float targets

**Usage**

```
RegressionBlock(n_out = NULL)
```

**Arguments**

n\_out            number of out features

**Value**

Block object

---

RemoveSilence	<i>Remove Silence</i>
---------------	-----------------------

---

**Description**

Split signal at points of silence greater than 2\*pad\_ms

**Usage**

```
RemoveSilence(
  remove_type = RemoveType()$Trim$value,
  threshold = 20,
  pad_ms = 20
)
```

**Arguments**

remove_type	remove type from RemoveType module
threshold	threshold point
pad_ms	pad milliseconds

**Value**

None

---

RemoveType	<i>RemoveType module</i>
------------	--------------------------

---

**Description**

RemoveType module

**Usage**

```
RemoveType()
```

**Value**

None



---

replace_all_caps	<i>Replace_all_caps</i>
------------------	-------------------------

---

**Description**

Replace tokens in ALL CAPS by their lower version and add 'TK\_UP' before.

**Usage**

```
replace_all_caps(t)
```

**Arguments**

t	text
---	------

**Value**

string

---

replace_maj	<i>Replace_maj</i>
-------------	--------------------

---

**Description**

Replace tokens in ALL CAPS by their lower version and add 'TK\_UP' before.

**Usage**

```
replace_maj(t)
```

**Arguments**

t	text
---	------

**Value**

string

replace\_rep

*Replace\_rep*

---

**Description**

Replace repetitions at the character level: cccc – TK\_REP 4 c

**Usage**

replace\_rep(t)

**Arguments**

t                    text

**Value**

string

---

replace\_wrep

*Replace\_wrep*

---

**Description**

Replace word repetitions: word word word word – TK\_WREP 4 word

**Usage**

replace\_wrep(t)

**Arguments**

t                    text

**Value**

string

---

Resample	<i>Resample</i>
----------	-----------------

---

**Description**

Resample using faster polyphase technique and avoiding FFT computation

**Usage**

```
Resample(sr_new)
```

**Arguments**

sr_new	input
--------	-------

**Value**

None

---

ResBlock	<i>ResBlock</i>
----------	-----------------

---

**Description**

Resnet block from 'ni' to 'nh' with 'stride'

**Usage**

```
ResBlock(  
  expansion,  
  ni,  
  nf,  
  stride = 1,  
  groups = 1,  
  reduction = NULL,  
  nh1 = NULL,  
  nh2 = NULL,  
  dw = FALSE,  
  g2 = 1,  
  sa = FALSE,  
  sym = FALSE,  
  norm_type = 1,  
  act_cls = nn$ReLU,  
  ndim = 2,  
  ks = 3,  
  pool = AvgPool(),
```

```

    pool_first = TRUE,
    padding = NULL,
    bias = NULL,
    bn_1st = TRUE,
    transpose = FALSE,
    init = "auto",
    xtra = NULL,
    bias_std = 0.01,
    dilation = 1,
    padding_mode = "zeros"
)

```

### Arguments

expansion	decoder
ni	number of linear inputs
nf	number of features
stride	stride number
groups	groups number
reduction	reduction
nh1	out channels 1
nh2	out channels 2
dw	dw paramer
g2	g2 block
sa	sa parameter
sym	symmetric
norm_type	normalization type
act_cls	activation
ndim	dimension number
ks	kernel size
pool	pooling type, Average, Max
pool_first	pooling first
padding	padding
bias	bias
bn_1st	batch normalization 1st
transpose	transpose
init	initializer
xtra	xtra
bias_std	bias standard deviation
dilation	dilation number
padding_mode	padding mode

**Value**

Block object

---

reshape	<i>Reshape</i>
---------	----------------

---

**Description**

resize x to (w,h)

**Usage**

```
reshape(x, h, w, resample = 0)
```

**Arguments**

x	tensor
h	height
w	width
resample	resample value

**Value**

None

---

Resize	<i>Resize</i>
--------	---------------

---

**Description**A transform that before\_call its state at each '`__call__`'**Usage**

```
Resize(size, method = "crop", pad_mode = "reflection", resamples = list(2, 0))
```

**Arguments**

size	size of image
method	method
pad_mode	reflection, zeros, border as string parameter
resamples	list of integers

**Value**

None

---

 ResizeBatch

*ResizeBatch*


---

**Description**

Reshape x to size, keeping batch dim the same size

**Usage**

```
ResizeBatch(...)
```

**Arguments**

... parameters to pass

**Value**

None

---

ResizeSignal

*Resize Signal*


---

**Description**

Crops signal to be length specified in ms by duration, padding if needed

**Usage**

```
ResizeSignal(duration, pad_mode = AudioPadType()$Zeros)
```

**Arguments**

duration int, duration  
 pad\_mode padding mode

**Value**

None

---

resize_max	<i>Resize_max</i>
------------	-------------------

---

**Description**

'resize' 'x' to 'max\_px', or 'max\_h', or 'max\_w'

**Usage**

```
resize_max(img, resample = 0, max_px = NULL, max_h = NULL, max_w = NULL)
```

**Arguments**

img	image
resample	resample value
max_px	max px
max_h	max height
max_w	max width

**Value**

None

---

ResNet	<i>ResNet</i>
--------	---------------

---

**Description**

Base class for all neural network modules.

**Usage**

```
ResNet(
  block,
  layers,
  num_classes = 1000,
  zero_init_residual = FALSE,
  groups = 1,
  width_per_group = 64,
  replace_stride_with_dilation = NULL,
  norm_layer = NULL
)
```

**Arguments**

block	the blocks that need to be passed to ResNet
layers	the layers to pass to ResNet
num_classes	the number of classes
zero_init_residual	logical, initializer
groups	the groups
width_per_group	the width per group
replace_stride_with_dilation	logical, replace stride with dilation
norm_layer	norm_layer

---

 resnet101

*Resnet101*


---

**Description**

ResNet-101 model from

**Usage**

```
resnet101(pretrained = FALSE, progress)
```

**Arguments**

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

"Deep Residual Learning for Image Recognition" <<https://arxiv.org/pdf/1512.03385.pdf>>

**Value**

model



---

resnet152

*Resnet152*

---

**Description**

Resnet152

**Usage**

```
resnet152(pretrained = FALSE, progress)
```

**Arguments**

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

"Deep Residual Learning for Image Recognition" <<https://arxiv.org/pdf/1512.03385.pdf>>

**Value**

model

---

resnet18

*Resnet18*

---

**Description**

Resnet18

**Usage**

```
resnet18(pretrained = FALSE, progress)
```

**Arguments**

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

"Deep Residual Learning for Image Recognition" <<https://arxiv.org/pdf/1512.03385.pdf>>

**Value**

model

---

resnet34	<i>Resnet34</i>
----------	-----------------

---

**Description**

ResNet-34 model from

**Usage**

```
resnet34(pretrained = FALSE, progress)
```

**Arguments**

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

"Deep Residual Learning for Image Recognition" <<https://arxiv.org/pdf/1512.03385.pdf>>

**Value**

model

---

resnet50	<i>Resnet50</i>
----------	-----------------

---

**Description**

Resnet50

**Usage**

```
resnet50(pretrained = FALSE, progress)
```

**Arguments**

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

"Deep Residual Learning for Image Recognition" <<https://arxiv.org/pdf/1512.03385.pdf>>

**Value**

model

---

ResnetBlock	<i>ResnetBlock</i>
-------------	--------------------

---

**Description**

nn()\$Module for the ResNet Block

**Usage**

```
ResnetBlock(
  dim,
  pad_mode = "reflection",
  norm_layer = NULL,
  dropout = 0,
  bias = TRUE
)
```

**Arguments**

dim	dimension
pad_mode	padding mode
norm_layer	normalization layer
dropout	dropout rate
bias	bias or not

**Value**

None

---

resnet_generator	<i>Resnet_generator</i>
------------------	-------------------------

---

**Description**

Resnet\_generator

**Usage**

```
resnet_generator(
  ch_in,
  ch_out,
  n_ftrs = 64,
  norm_layer = NULL,
  dropout = 0,
  n_blocks = 9,
  pad_mode = "reflection"
)
```

**Arguments**

ch_in	input
ch_out	output
n_ftrs	filter
norm_layer	normalization layer
dropout	dropout rate
n_blocks	number of blocks
pad_mode	padding mode

**Value**

None

---

res_block_1d	<i>Res_block_1d</i>
--------------	---------------------

---

**Description**

Resnet block as described in the paper.

**Usage**

```
res_block_1d(nf, ks = c(5, 3))
```

**Arguments**

nf	number of features
ks	kernel size

**Value**

block

---

RetinaNet

*RetinaNet*

---

### Description

Implements RetinaNet from <https://arxiv.org/abs/1708.02002>

### Usage

```
RetinaNet(...)
```

### Arguments

```
... arguments to pass
```

### Value

```
model
```

### Examples

```
## Not run:  
  
encoder = create_body(resnet34(), pretrained = TRUE)  
arch = RetinaNet(encoder, get_c(dls), final_bias=-4)  
  
## End(Not run)
```

---

RetinaNetFocalLoss

*RetinaNetFocalLoss*

---

### Description

Base class for all neural network modules.

### Usage

```
RetinaNetFocalLoss(...)
```

### Arguments

```
... parameters to pass
```

**Details**

Your models should also subclass this class. Modules can also contain other Modules, allowing to nest them in a tree structure. You can assign the submodules as regular attributes::  
`import torch.nn as nn import torch.nn.functional as F class Model(nn.Module): def __init__(self): super(Model, self).__init__() self.conv1 = nn.Conv2d(1, 20, 5) self.conv2 = nn.Conv2d(20, 20, 5) def forward(self, x): x = F.relu(self.conv1(x)) return F.relu(self.conv2(x))`  
 Submodules assigned in this way will be registered, and will have their parameters converted too when you call `:meth:'to'`, etc.

**Value**

None

---

retinanet_	<i>Retinanet module</i>
------------	-------------------------

---

**Description**

Retinanet module

**Usage**`retinanet_()`**Value**

None

---

reverse_text	<i>Reverse_text</i>
--------------	---------------------

---

**Description**

Reverse\_text

**Usage**`reverse_text(x)`**Arguments**

x	text
---	------

**Value**

string

---

`rgb2hsv`*Rgb2hsv*

---

**Description**

Converts a RGB image to an HSV image.

**Usage**

```
rgb2hsv(img)
```

**Arguments**

`img`            image object

**Details**

Note: Will not work on logit space images.

**Value**

None

---

`rmse`*RMSE*

---

**Description**

Root mean squared error

**Usage**

```
rmse(preds, targs)
```

**Arguments**

`preds`            predictions  
`targs`            targets

**Value**

None

**Examples**

```
## Not run:

model = dls %>% tabular_learner(layers=c(200,100,100,200),
metrics = list(mse(),rmse()) )

## End(Not run)
```

---

RMSProp

*RMSProp*


---

**Description**

RMSProp

**Usage**

RMSProp(...)

**Arguments**

... parameters to pass

**Value**

None

---

rms\_prop\_step

*Rms\_prop\_step*


---

**Description**

Step for SGD with momentum with 'lr'

**Usage**

rms\_prop\_step(p, lr, sqr\_avg, eps, grad\_avg = NULL, ...)



**Arguments**

p	p
lr	learning rate
sqr_avg	sqr average
eps	epsilon
grad_avg	grad average
...	additional arguments to pass

**Value**

None

---

*rm\_useless\_spaces*      *Rm\_useless\_spaces*

---

**Description**

Remove multiple spaces

**Usage**

`rm_useless_spaces(t)`

**Arguments**

t                      text

**Value**

string

**Examples**

```
## Not run:  
  
rm_useless_spaces('hello,  Sir!')  
  
## End(Not run)
```

---

RNNDropout

*RNNDropout*


---

**Description**

Dropout with probability ‘p’ that is consistent on the seq\_len dimension.

**Usage**

```
RNNDropout(p = 0.5)
```

**Arguments**

p                    p

**Value**

None

---

RNNRegularizer

*RNNRegularizer*


---

**Description**

‘Callback’ that adds AR and TAR regularization in RNN training

**Usage**

```
RNNRegularizer(alpha = 0, beta = 0)
```

**Arguments**

alpha                alpha  
beta                  beta

**Value**

None

---

RocAuc	<i>RocAuc</i>
--------	---------------

---

**Description**

Area Under the Receiver Operating Characteristic Curve for single-label multiclass classification problems

**Usage**

```
RocAuc(  
  axis = -1,  
  average = "macro",  
  sample_weight = NULL,  
  max_fpr = NULL,  
  multi_class = "ovr"  
)
```

**Arguments**

axis	axis
average	average
sample_weight	sample_weight
max_fpr	max_fpr
multi_class	multi_class

**Value**

None

---

RocAucBinary	<i>RocAucBinary</i>
--------------	---------------------

---

**Description**

Area Under the Receiver Operating Characteristic Curve for single-label binary classification problems

**Usage**

```
RocAucBinary(  
  axis = -1,  
  average = "macro",  
  sample_weight = NULL,  
  max_fpr = NULL,  
  multi_class = "raise"  
)
```

**Arguments**

axis	axis
average	average
sample_weight	sample_weight
max_fpr	max_fpr
multi_class	multi_class

**Value**

None

**Examples**

```
## Not run:

model = dls %>% tabular_learner(layers=c(200,100,100,200),
  config = tabular_config(embed_p = 0.3, use_bn = FALSE),
  metrics = list(accuracy, RocAucBinary(),
                Precision(), Recall(),
                F1Score()))

## End(Not run)
```

---

RocAucMulti

*RocAucMulti*

---

**Description**

Area Under the Receiver Operating Characteristic Curve for multi-label binary classification problems

**Usage**

```
RocAucMulti(
  sigmoid = TRUE,
  average = "macro",
  sample_weight = NULL,
  max_fpr = NULL
)
```

**Arguments**

sigmoid	sigmoid
average	average
sample_weight	sample_weight
max_fpr	max_fpr

**Value**

None

---

Rotate	<i>Rotate</i>
--------	---------------

---

**Description**

Apply a random rotation of at most 'max\_deg' with probability 'p' to a batch of images

**Usage**

```
Rotate(
    max_deg = 10,
    p = 0.5,
    draw = NULL,
    size = NULL,
    mode = "bilinear",
    pad_mode = "reflection",
    align_corners = TRUE,
    batch = FALSE
)
```

**Arguments**

max_deg	maximum degrees
p	probability
draw	draw
size	size of image
mode	mode
pad_mode	reflection, zeros, border as string parameter
align_corners	align corners or not
batch	batch or not

**Value**

None

---

rotate_mat	<i>Rotate_mat</i>
------------	-------------------

---

**Description**

Return a random rotation matrix with 'max\_deg' and 'p'

**Usage**

```
rotate_mat(x, max_deg = 10, p = 0.5, draw = NULL, batch = FALSE)
```

**Arguments**

x	tensor
max_deg	max_deg
p	probability
draw	draw
batch	batch

**Value**

None

---

round	<i>Round</i>
-------	--------------

---

**Description**

Round

**Usage**

```
## S3 method for class 'torch.Tensor'
round(x, digits = 0)
```

**Arguments**

x	tensor
digits	decimal

**Value**

tensor

---

round.fastai.torch\_core.TensorMask  
*Round*

---

**Description**

Round

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
round(x, digits = 0)
```

**Arguments**

x	tensor
digits	decimal

**Value**

tensor

---

Saturation                      *Saturation*

---

**Description**

Apply change in saturation of 'max\_lighting' to batch of images with probability 'p'.

**Usage**

```
Saturation(max_lighting = 0.2, p = 0.75, draw = NULL, batch = FALSE)
```

**Arguments**

max_lighting	maximum lighting
p	probability
draw	draw
batch	batch

**Value**

None

---

SaveModelCallback	<i>SaveModelCallback</i>
-------------------	--------------------------

---

**Description**

SaveModelCallback

**Usage**

SaveModelCallback(...)

**Arguments**

... parameters to pass

**Value**

None

---

SchedCos	<i>SchedCos</i>
----------	-----------------

---

**Description**

Cosine schedule function from 'start' to 'end'

**Usage**

SchedCos(start, end)

**Arguments**

start	start
end	end

**Value**

None



---

SchedExp	<i>SchedExp</i>
----------	-----------------

---

**Description**

Exponential schedule function from 'start' to 'end'

**Usage**

SchedExp(start, end)

**Arguments**

start	start
end	end

**Value**

None

---

SchedLin	<i>SchedLin</i>
----------	-----------------

---

**Description**

Linear schedule function from 'start' to 'end'

**Usage**

SchedLin(start, end)

**Arguments**

start	start
end	end

**Value**

None

SchedNo                      *SchedNo*

---

**Description**

Constant schedule function with 'start' value

**Usage**

SchedNo(start, end)

**Arguments**

start	start
end	end

**Value**

None

---

SchedPoly                      *SchedPoly*

---

**Description**

Polynomial schedule (of 'power') function from 'start' to 'end'

**Usage**

SchedPoly(start, end, power)

**Arguments**

start	start
end	end
power	power

**Value**

None

---

 SEBlock

*SEBlock*


---

**Description**

SEBlock

**Usage**

```
SEBlock(expansion, ni, nf, groups = 1, reduction = 16, stride = 1)
```

**Arguments**

expansion	decoder
ni	number of inputs
nf	number of features
groups	number of groups
reduction	number of reduction
stride	number of strides

**Value**

Block object

---

 SegmentationDataLoaders\_from\_label\_func

*SegmentationDataLoaders\_from\_label\_func*


---

**Description**

Create from list of 'fnames' in 'path's with 'label\_func'.

**Usage**

```
SegmentationDataLoaders_from_label_func(
  path,
  fnames,
  label_func,
  valid_pct = 0.2,
  seed = NULL,
  codes = NULL,
  item_tfms = NULL,
  batch_tfms = NULL,
  bs = 64,
```

```

    val_bs = NULL,
    shuffle_train = TRUE,
    device = NULL
)

```

### Arguments

path	path
fnames	file names
label_func	label function
valid_pct	validation percentage
seed	seed
codes	codes
item_tfms	item transformations
batch_tfms	batch transformations
bs	batch size
val_bs	validation batch size
shuffle_train	shuffle train
device	device name

### Value

None

---

SelfAttention	<i>SelfAttention</i>
---------------	----------------------

---

### Description

Self attention layer for ‘n\_channels’.

### Usage

```
SelfAttention(n_channels)
```

### Arguments

n_channels	number of channels
------------	--------------------

### Value

None

---

SEModule	<i>SEModule</i>
----------	-----------------

---

**Description**

SEModule

**Usage**

```
SEModule(ch, reduction, act_cls = nn()$ReLU)
```

**Arguments**

ch	ch
reduction	reduction
act_cls	activation

**Value**

None

---

SentenceEncoder	<i>SentenceEncoder</i>
-----------------	------------------------

---

**Description**

Create an encoder over ‘module’ that can process a full sentence.

**Usage**

```
SentenceEncoder(bptt, module, pad_idx = 1, max_len = NULL)
```

**Arguments**

bptt	bptt
module	module
pad_idx	pad_idx
max_len	max_len

**Value**

None

SentencePieceTokenizer

*SentencePieceTokenizer*

---

**Description**

SentencePiece tokenizer for 'lang'

**Usage**

```
SentencePieceTokenizer(  
    lang = "en",  
    special_toks = NULL,  
    sp_model = NULL,  
    vocab_sz = NULL,  
    max_vocab_sz = 30000,  
    model_type = "unigram",  
    char_coverage = NULL,  
    cache_dir = "tmp"  
)
```

**Arguments**

lang	lang
special_toks	special_toks
sp_model	sp_model
vocab_sz	vocab_sz
max_vocab_sz	max_vocab_sz
model_type	model_type
char_coverage	char_coverage
cache_dir	cache_dir

**Value**

None

---

SeparableBlock	<i>SeparableBlock</i>
----------------	-----------------------

---

**Description**

SeparableBlock

**Usage**

```
SeparableBlock(expansion, ni, nf, reduction = 16, stride = 1, base_width = 4)
```

**Arguments**

expansion	decoder
ni	number of inputs
nf	number of features
reduction	number of reduction
stride	number of stride
base_width	base width

**Value**

Block object

---

sequential	<i>Sequential</i>
------------	-------------------

---

**Description**

Sequential

**Usage**

```
sequential(...)
```

**Arguments**

...	parameters to pass
-----	--------------------

**Value**

None

---

SequentialEx

*SequentialEx*

---

**Description**

SequentialEx

**Usage**

SequentialEx(...)

**Arguments**

... parameters to pass

**Value**

None

---

SequentialRNN

*Sequential RNN*

---

**Description**

Sequential RNN

**Usage**

SequentialRNN(...)

**Arguments**

... parameters to pass

**Value**

layer



---

SEResNeXtBlock	<i>SEResNeXtBlock</i>
----------------	-----------------------

---

**Description**

SEResNeXtBlock

**Usage**

```
SEResNeXtBlock(
    expansion,
    ni,
    nf,
    groups = 32,
    reduction = 16,
    stride = 1,
    base_width = 4
)
```

**Arguments**

expansion	decoder
ni	number of linear inputs
nf	number of features
groups	groups number
reduction	reduction number
stride	stride number
base_width	int, base width

**Value**

Block object

---

setup_aug_tfms	<i>Setup_aug_tfms</i>
----------------	-----------------------

---

**Description**

Go through 'tfms' and combines together affine/coord or lighting transforms

**Usage**

```
setup_aug_tfms(tfms)
```

**Arguments**

tfms	transformations
------	-----------------

**Value**

None

---

set_freeze_model	<i>Set freeze model</i>
------------------	-------------------------

---

**Description**

Set freeze model

**Usage**

```
set_freeze_model(m, rg)
```

**Arguments**

m	parameters
rg	rg

**Value**

None

---

set_item_pg	<i>Set_item_pg</i>
-------------	--------------------

---

**Description**

Set\_item\_pg

**Usage**

```
set_item_pg(pg, k, v)
```

**Arguments**

pg	pg
k	k
v	v

**Value**

None

---

SGD

*SGD*

---

**Description**

SGD

**Usage**

SGD(...)

**Arguments**

... parameters to pass

**Value**

None

---

sgd\_step

*Sgd\_step*

---

**Description**

Sgd\_step

**Usage**

sgd\_step(p, lr, ...)

**Arguments**

p p  
lr learning rate  
... additional arguments to pass

**Value**

None

## Examples

```
## Not run:

tst_param = function(val, grad = NULL) {
  "Create a tensor with `val` and a gradient of `grad` for testing"
  res = tensor(val) %>% float()

  if(is.null(grad)) {
    grad = tensor(val / 10)
  } else {
    grad = tensor(grad)
  }

  res$grad = grad %>% float()
  res
}
p = tst_param(1., 0.1)
sgd_step(p, 1.)

## End(Not run)
```

---

SGRoll

*SGRoll*

---

## Description

Shifts spectrogram along x-axis wrapping around to other side

## Usage

```
SGRoll(max_shift_pct = 0.5, direction = 0)
```

## Arguments

max_shift_pct	maximum shift percentage
direction	direction

## Value

None

---

shap	<i>Shap module</i>
------	--------------------

---

**Description**

Shap module

**Usage**

shap()

**Value**

None

---

shape	<i>Shape</i>
-------	--------------

---

**Description**

Shape

**Usage**

shape(img)

**Arguments**

img	image
-----	-------

**Value**

None

---

 ShapInterpretation      *ShapInterpretation*


---

### Description

Base interpreter to use the 'SHAP' interpretation library

### Usage

```
ShapInterpretation(
    learn,
    test_data = NULL,
    link = "identity",
    l1_reg = "auto",
    n_samples = 128
)
```

### Arguments

learn	learner/model
test_data	should be either a Pandas dataframe or a TabularDataLoader. If not, 100 random rows of the training data will be used instead.
link	link can either be "identity" or "logit". A generalized linear model link to connect the feature importance values to the model output. Since the feature importance values, phi, sum up to the model output, it often makes sense to connect them to the output with a link function where $\text{link}(\text{outout}) = \text{sum}(\text{phi})$ . If the model output is a probability then the LogitLink link function makes the feature importance values have log-odds units.
l1_reg	can be an integer value representing the number of features, "auto", "aic", "bic", or a float value. The l1 regularization to use for feature selection (the estimation procedure is based on a debiased lasso). The auto option currently uses "aic" when less than 20 space is enumerated, otherwise it uses no regularization.
n_samples	can either be "auto" or an integer value. This is the number of times to re-evaluate the model when explaining each predictions. More samples leads to lower variance estimations of the SHAP values

### Value

None

---

Shortcut	<i>Shortcut</i>
----------	-----------------

---

**Description**

Merge a shortcut with the result of the module by adding them. Adds Conv, BN and ReLU

**Usage**

```
Shortcut(ni, nf, act_fn = nn$ReLU(inplace = TRUE))
```

**Arguments**

ni	number of input channels
nf	number of features
act_fn	activation

**Value**

None

---

ShortEpochCallback	<i>ShortEpochCallback</i>
--------------------	---------------------------

---

**Description**

Fit just 'pct' of an epoch, then stop

**Usage**

```
ShortEpochCallback(pct = 0.01, short_valid = TRUE)
```

**Arguments**

pct	percentage
short_valid	short_valid or not

**Value**

None

---

show	<i>Show</i>
------	-------------

---

**Description**

Adds functionality to view dicom images where each file may have more than 1 frame

**Usage**

```
show(img, frames = 1, scale = TRUE, ...)
```

**Arguments**

img	image object
frames	number of frames
scale	scale
...	additional arguments

**Value**

None

---

ShowCycleGANImgsCallback	<i>ShowCycleGANImgsCallback</i>
--------------------------	---------------------------------

---

**Description**

Update the progress bar with input and prediction images

**Usage**

```
ShowCycleGANImgsCallback(imgA = FALSE, imgB = TRUE, show_img_interval = 10)
```

**Arguments**

imgA	img from A domain
imgB	img from B domain
show_img_interval	show image interval

**Value**

None



---

ShowGraphCallback	<i>ShowGraphCallback</i>
-------------------	--------------------------

---

**Description**

ShowGraphCallback

**Usage**

```
ShowGraphCallback(...)
```

**Arguments**

... parameters to pass

**Value**

None

---

show_array	<i>Show_array</i>
------------	-------------------

---

**Description**

Show an array on 'ax'.

**Usage**

```
show_array(
    array,
    ax = NULL,
    figsize = NULL,
    title = NULL,
    ctx = NULL,
    tx = NULL
)
```

**Arguments**

array	R array
ax	axis
figsize	figure size
title	title, text
ctx	ctx
tx	tx

**Value**

None

**Examples**

```
## Not run:

arr = as.array(1:10)
show_array(arr,title = 'My R array') %>% plot(dpi = 200)

## End(Not run)
```

---

show\_batch

*Show\_batch*


---

**Description**

Show\_batch

**Usage**

```
show_batch(
  dls,
  b = NULL,
  max_n = 9,
  ctxs = NULL,
  figsize = c(6, 6),
  show = TRUE,
  unique = FALSE,
  dpi = 120,
  ...
)
```

**Arguments**

dls	dataloader object
b	defaults to one_batch
max_n	maximum images
ctxs	ctxs parameter
figsize	figure size
show	show or not

unique	unique images
dpi	dots per inch
...	additional arguments to pass

**Value**

None

**Examples**

```
## Not run:  
  
dls %>% show_batch()  
  
## End(Not run)
```

---

`show_image`*Show\_image*

---

**Description**

Show a PIL or PyTorch image on 'ax'.

**Usage**

```
show_image(  
  im,  
  ax = NULL,  
  figsize = NULL,  
  title = NULL,  
  ctx = NULL,  
  cmap = NULL,  
  norm = NULL,  
  aspect = NULL,  
  interpolation = NULL,  
  alpha = NULL,  
  vmin = NULL,  
  vmax = NULL,  
  origin = NULL,  
  extent = NULL  
)
```

**Arguments**

im	im
ax	axis
figsize	figure size
title	title
ctx	ctx
cmap	color maps
norm	normalization
aspect	aspect
interpolation	interpolation
alpha	alpha value
vmin	value min
vmax	value max
origin	origin
extent	extent

---

 show\_images

*Show\_images*


---

**Description**

Show all images 'ims' as subplots with 'rows' using 'titles'

**Usage**

```
show_images(
    ims,
    nrows = 1,
    ncols = NULL,
    titles = NULL,
    figsize = NULL,
    imsize = 3,
    add_vert = 0
)
```

**Arguments**

ims	images
nrows	number of rows
ncols	number of columns
titles	titles
figsize	figure size
imsize	image size
add_vert	add vertical

**Value**

None

---

`show_preds`

---

*Show\_preds*

---

**Description**

Show\_preds

**Usage**

```

show_preds(
    predictions,
    idx,
    class_map = NULL,
    denormalize_fn = denormalize_imagenet(),
    display_label = TRUE,
    display_bbox = TRUE,
    display_mask = TRUE,
    ncols = 1,
    figsize = NULL,
    show = FALSE,
    dpi = 100
)

```

**Arguments**

<code>predictions</code>	provide list of raw predictions
<code>idx</code>	image indices
<code>class_map</code>	<code>class_map</code>
<code>denormalize_fn</code>	<code>denormalize_fn</code>
<code>display_label</code>	<code>display_label</code>
<code>display_bbox</code>	<code>display_bbox</code>
<code>display_mask</code>	<code>display_mask</code>
<code>ncols</code>	<code>ncols</code>
<code>figsize</code>	<code>figsize</code>
<code>show</code>	<code>show</code>
<code>dpi</code>	dots per inch

**Value**

None

---

show_results	<i>Show_results</i>
--------------	---------------------

---

**Description**

Show some predictions on 'ds\_idx'-th dataset or 'dl'

**Usage**

```
show_results(
    object,
    ds_idx = 1,
    dl = NULL,
    max_n = 9,
    shuffle = TRUE,
    dpi = 90,
    ...
)
```

**Arguments**

object	model
ds_idx	ds by index
dl	dataloader
max_n	maximum number of images
shuffle	shuffle or not
dpi	dots per inch
...	additional arguments

**Value**

None

---

show_samples	<i>Show_samples</i>
--------------	---------------------

---

**Description**

Show\_samples

**Usage**

```

show_samples(
    dls,
    idx,
    class_map = NULL,
    denormalize_fn = denormalize_imagenet(),
    display_label = TRUE,
    display_bbox = TRUE,
    display_mask = TRUE,
    ncols = 1,
    figsize = NULL,
    show = FALSE,
    dpi = 100
)

```

**Arguments**

dls	dataloader
idx	image indices
class_map	class_map
denormalize_fn	denormalize_fn
display_label	display_label
display_bbox	display_bbox
display_mask	display_mask
ncols	ncols
figsize	figsize
show	show
dpi	dots per inch

**Value**

None

---

sigmoid

*Sigmoid*

---

**Description**

Same as ‘torch\$sigmoid’, plus clamping to ‘(eps,1-eps)’

**Usage**

```
sigmoid(input, eps = 1e-07)
```

**Arguments**

input	inputs
eps	epsilon

**Value**

None

SigmoidRange

*SigmoidRange***Description**

Sigmoid module with range '(low, high)'

**Usage**

SigmoidRange(low, high)

**Arguments**

low	low value
high	high value

**Value**

None

sigmoid\_

*Sigmoid\_***Description**

Same as 'torch\$sigmoid\_', plus clamping to '(eps,1-eps)

**Usage**

sigmoid\_(input, eps = 1e-07)

**Arguments**

input	input
eps	eps

**Value**

None



---

sigmoid_range	<i>Sigmoid_range</i>
---------------	----------------------

---

**Description**

Sigmoid function with range '(low, high)'

**Usage**

```
sigmoid_range(x, low, high)
```

**Arguments**

x	tensor
low	low value
high	high value

**Value**

None

---

SignalCutout	<i>Signal Cutout</i>
--------------	----------------------

---

**Description**

Randomly zeros some portion of the signal

**Usage**

```
SignalCutout(p = 0.5, max_cut_pct = 0.15)
```

**Arguments**

p	probability
max_cut_pct	max cut percentage

**Value**

None

---

 SignalLoss
 

---

*Signal Loss***Description**

Randomly loses some portion of the signal

**Usage**

```
SignalLoss(p = 0.5, max_loss_pct = 0.15)
```

**Arguments**

p	probability
max_loss_pct	max loss percentage

**Value**

None

---

SignalShifter

*Signal Shifter***Description**

Randomly shifts the audio signal by 'max\_pct'

**Usage**

```
SignalShifter(
  p = 0.5,
  max_pct = 0.2,
  max_time = NULL,
  direction = 0,
  roll = FALSE
)
```

**Arguments**

p	probability
max_pct	max percentage
max_time	maximum time
direction	direction
roll	roll or not

**Details**

direction must be -1(left) 0(bidirectional) or 1(right).

**Value**

None

---

SimpleCNN

*SimpleCNN*

---

**Description**

Create a simple CNN with 'filters'.

**Usage**

```
SimpleCNN(filters, kernel_szs = NULL, strides = NULL, bn = TRUE)
```

**Arguments**

filters	filters number
kernel_szs	kernel size
strides	strides
bn	batch normalization

**Value**

None

---

SimpleSelfAttention

*SimpleSelfAttention*

---

**Description**

Same as 'nn()\$Module', but no need for subclasses to call 'super()\$\_\_init\_\_\$'

**Usage**

```
SimpleSelfAttention(n_in, ks = 1, sym = FALSE)
```

**Arguments**

n_in	inputs
ks	kernel size
sym	sym

**Value**

None

---

sin.fastai.torch\_core.TensorMask  
*Sin*

---

**Description**

Sin

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
sin(x)
```

**Arguments**

x                    tensor

**Value**

tensor

---

sinh.fastai.torch\_core.TensorMask  
*Sinh*

---

**Description**

Sinh

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
sinh(x)
```

**Arguments**

x                    tensor

**Value**

tensor

---

sin_	<i>Sin</i>
------	------------

---

**Description**

Sin

**Usage**

```
## S3 method for class 'torch.Tensor'
sin(x)
```

**Arguments**

x	tensor
---	--------

**Value**

tensor

---

skm_to_fastai	<i>Skm to fastai</i>
---------------	----------------------

---

**Description**

Convert 'func' from sklearn\$metrics to a fastai metric

**Usage**

```
skm_to_fastai(
  func,
  is_class = TRUE,
  thresh = NULL,
  axis = -1,
  activation = NULL,
  ...
)
```

**Arguments**

func	function
is_class	is classification or not
thresh	threshold point
axis	axis
activation	activation
...	additional arguments to pass

**Value**

None

---

`slice`*Slice*

---

**Description**

Slice

**Usage**`slice(...)`**Arguments**`...` additional arguments**Details**`slice(start, stop[, step])` Create a slice object. This is used for extended slicing (e.g. `a[0:10:2]`).**Value**

sliced object

---

`sort`*Sort*

---

**Description**

Sort

**Usage**

```
## S3 method for class 'torch.Tensor'
sort(x, decreasing = FALSE, ...)
```

**Arguments**

<code>x</code>	tensor
<code>decreasing</code>	the order
<code>...</code>	additional parameters to pass

---

```
sort.fastai.torch_core.TensorMask  
Sort
```

---

**Description**

Sort

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
sort(x, decreasing = FALSE, ...)
```

**Arguments**

x	tensor
decreasing	the order
...	additional parameters to pass

**Value**

tensor

---

```
SortedDL          SortedDL
```

---

**Description**

A ‘DataLoader’ that goes through the item in the order given by ‘sort\_func’

**Usage**

```
SortedDL(  
  dataset,  
  sort_func = NULL,  
  res = NULL,  
  bs = 64,  
  shuffle = FALSE,  
  num_workers = NULL,  
  verbose = FALSE,  
  do_setup = TRUE,  
  pin_memory = FALSE,  
  timeout = 0,  
  batch_size = NULL,  
  drop_last = FALSE,
```

```

    indexed = NULL,
    n = NULL,
    device = NULL
)

```

### Arguments

dataset	dataset
sort_func	sort_func
res	res
bs	bs
shuffle	shuffle
num_workers	num_workers
verbose	verbose
do_setup	do_setup
pin_memory	pin_memory
timeout	timeout
batch_size	batch_size
drop_last	drop_last
indexed	indexed
n	n
device	device

### Value

None

---

SpacyTokenizer	<i>SpacyTokenizer</i>
----------------	-----------------------

---

### Description

Spacy tokenizer for 'lang'

### Usage

```
SpacyTokenizer(lang = "en", special_toks = NULL, buf_sz = 5000)
```

### Arguments

lang	language
special_toks	special tokenizers
buf_sz	buffer size



**Value**

none

---

SpearmanCorrCoef	<i>SpearmanCorrCoef</i>
------------------	-------------------------

---

**Description**

Spearman correlation coefficient for regression problem

**Usage**

```
SpearmanCorrCoef(
    dim_argmax = NULL,
    axis = 0,
    nan_policy = "propagate",
    activation = "no",
    thresh = NULL,
    to_np = FALSE,
    invert_arg = FALSE,
    flatten = TRUE
)
```

**Arguments**

dim_argmax	dim_argmax
axis	axis
nan_policy	nan_policy
activation	activation
thresh	thresh
to_np	to_np
invert_arg	invert_arg
flatten	flatten

**Value**

None

---

SpectrogramTransformer

*Spectrogram Transformer*

---

### Description

Creates a factory for creating AudioToSpec

### Usage

SpectrogramTransformer(mel = TRUE, to\_db = TRUE)

### Arguments

mel	mel-spectrogram or not
to_db	to decibels

### Details

transforms with different parameters

### Value

None

---

spec\_add\_spaces

*Spec\_add\_spaces*

---

### Description

Add spaces around / and #

### Usage

spec\_add\_spaces(t)

### Arguments

t	text
---	------

### Value

string

---

sqrD

*SqrD*

---

### Description

SqrD

### Usage

```
## S3 method for class 'torch.Tensor'  
sqrD(x)
```

### Arguments

x            tensor

### Value

tensor

---

sqrD.fastai.torch\_core.TensorMask

*SqrD*

---

### Description

SqrD

### Usage

```
## S3 method for class 'fastai.torch_core.TensorMask'  
sqrD(x)
```

### Arguments

x            tensor

### Value

tensor

---

 SqueezeNet

*SqueezeNet*


---

### Description

Base class for all neural network modules.

### Usage

```
SqueezeNet(version = "1_0", num_classes = 1000)
```

### Arguments

version	version of SqueezeNet
num_classes	the number of classes

### Details

Your models should also subclass this class. Modules can also contain other Modules, allowing to nest them in a tree structure. You can assign the submodules as regular attributes::

```
import torch.nn as nn
import torch.nn.functional as F
class Model(nn.Module):
    def __init__(self):
        super(Model, self).__init__()
        self.conv1 = nn.Conv2d(1, 20, 5)
        self.conv2 = nn.Conv2d(20, 20, 5)
    def forward(self, x):
        x = F.relu(self.conv1(x))
        return F.relu(self.conv2(x))
```

Submodules assigned in this way will be registered, and will have their parameters converted too when you call :meth:`to`, etc.

### Value

model

---

 squeezeNet1\_0

*SqueezeNet1\_0*


---

### Description

SqueezeNet model architecture from the "SqueezeNet: AlexNet-level

### Usage

```
squeezeNet1_0(pretrained = FALSE, progress)
```

### Arguments

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

accuracy with 50x fewer parameters and <0.5MB model size" <<https://arxiv.org/abs/1602.07360>>' paper.

**Value**

model

---

squeezenet1_1	<i>Squeezenet1_1</i>
---------------	----------------------

---

**Description**

SqueezeNet 1.1 model from the 'official SqueezeNet repo

**Usage**

```
squeezenet1_1(pretrained = FALSE, progress)
```

**Arguments**

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

<[https://github.com/DeepScale/SqueezeNet/tree/master/SqueezeNet\\_v1.1](https://github.com/DeepScale/SqueezeNet/tree/master/SqueezeNet_v1.1)>'. SqueezeNet 1.1 has 2.4x less computation and slightly fewer parameters than SqueezeNet 1.0, without sacrificing accuracy.

**Value**

model

---

stack_train_valid	<i>Stack_train_valid</i>
-------------------	--------------------------

---

**Description**

Stack df\_train and df\_valid, adds 'valid\_col'=TRUE/FALSE for df\_valid/df\_train

**Usage**

```
stack_train_valid(df_train, df_valid)
```

**Arguments**

df_train	train data
df_valid	validation data

**Value**

data frame

---

step_stat	<i>Step_stat</i>
-----------	------------------

---

**Description**

Register the number of steps done in 'state' for 'p'

**Usage**

```
step_stat(p, step = 0, ...)
```

**Arguments**

p	p
step	step
...	additional args to pass

**Value**

None

---

sub	<i>Sub</i>
-----	------------

---

**Description**

Sub

**Usage**

```
## S3 method for class 'torch.Tensor'  
a - b
```

**Arguments**

a	tensor
b	tensor

**Value**

tensor

---

subplots	<i>Subplots</i>
----------	-----------------

---

**Description**

Subplots

**Usage**

```
subplots(nrows = 2, ncols = 2, figsize = NULL, imsize = 4)
```

**Arguments**

nrows	number of rows
ncols	number of columns
figsize	figure size
imsize	image size

**Value**

plot object

---

 sub\_mask
*Sub***Description**

Sub

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'
a - b
```

**Arguments**

a            tensor

b            tensor

**Value**

tensor

---

 summarization\_splitter
*Summarization\_splitter***Description**

Custom param splitter for summarization models

**Usage**

```
summarization_splitter(m, arch)
```

**Arguments**

m            splitter parameter

arch        architecture

**Value**

None



---

```
summary.fastai.learner.Learner
```

*Summary*

---

**Description**

Summary

**Usage**

```
## S3 method for class 'fastai.learner.Learner'  
summary(object, ...)
```

**Arguments**

object	model
...	additional arguments to pass

**Value**

None

**Examples**

```
## Not run:  
  
summary(model)  
  
## End(Not run)
```

---

```
summary.fastai.tabular.learner.TabularLearner
```

*Summary*

---

**Description**

Print a summary of 'm' using a output text width of 'n' chars

**Usage**

```
## S3 method for class 'fastai.tabular.learner.TabularLearner'  
summary(object, ...)
```

**Arguments**

object	model
...	additional parameters to pass

**Value**

None

---

summary_plot	<i>Summary_plot</i>
--------------	---------------------

---

**Description**

Displays the SHAP values (which can be interpreted for feature importance)

**Usage**

```
summary_plot(object, dpi = 200, ...)
```

**Arguments**

object	ShapInterpretation object
dpi	dots per inch
...	additional arguments

**Value**

None

---

swish	<i>Swish</i>
-------	--------------

---

**Description**

Swish

**Usage**

```
swish(x, inplace = FALSE)
```

**Arguments**

x	tensor
inplace	inplace or not

**Value**

None

---

`Swish_`*Swish*

---

**Description**

Same as `nn()$Module`, but no need for subclasses to call `super()$__init__`

**Usage**

```
Swish_(...)
```

**Arguments**

... parameters to pass

**Value**

None

---

`tabular`*Tabular*

---

**Description**

Tabular

**Usage**

```
tabular()
```

**Value**

None

---

TabularDataTable      *TabularDataTable*

---

### Description

A ‘Tabular’ object with transforms

### Usage

```
TabularDataTable(
  df,
  procs = NULL,
  cat_names = NULL,
  cont_names = NULL,
  y_names = NULL,
  y_block = NULL,
  splits = NULL,
  do_setup = TRUE,
  device = NULL,
  inplace = FALSE,
  reduce_memory = TRUE,
  ...
)
```

### Arguments

<code>df</code>	A DataFrame of your data
<code>procs</code>	list of preprocess functions
<code>cat_names</code>	the names of the categorical variables
<code>cont_names</code>	the names of the continuous variables
<code>y_names</code>	the names of the dependent variables
<code>y_block</code>	the TransformBlock to use for the target
<code>splits</code>	How to split your data
<code>do_setup</code>	A parameter for if Tabular will run the data through the procs upon initialization
<code>device</code>	cuda or cpu
<code>inplace</code>	If True, Tabular will not keep a separate copy of your original DataFrame in memory
<code>reduce_memory</code>	fastai will attempt to reduce the overall memory usage
<code>...</code>	additional parameters to pass

### Value

None

---

`TabularModel`*TabularModel*

---

**Description**

Basic model for tabular data.

**Usage**

```
TabularModel(  
  emb_szs,  
  n_cont,  
  out_sz,  
  layers,  
  ps = NULL,  
  embed_p = 0,  
  y_range = NULL,  
  use_bn = TRUE,  
  bn_final = FALSE,  
  bn_cont = TRUE,  
  act_cls = nn()$ReLU(inplace = TRUE)  
)
```

**Arguments**

<code>emb_szs</code>	embedding size
<code>n_cont</code>	number of cont
<code>out_sz</code>	output size
<code>layers</code>	layers
<code>ps</code>	ps
<code>embed_p</code>	embed proportion
<code>y_range</code>	y range
<code>use_bn</code>	use batch normalization
<code>bn_final</code>	batch normalization final
<code>bn_cont</code>	batch normalization cont
<code>act_cls</code>	activation

**Value**

None

---

 TabularTS

*TabularTS*


---

### Description

A 'DataFrame' wrapper that knows which cols are x/y, and returns rows in '`__getitem__`'

### Usage

```

TabularTS(
    df,
    procs = NULL,
    x_names = NULL,
    y_names = NULL,
    block_y = NULL,
    splits = NULL,
    do_setup = TRUE,
    device = NULL,
    inplace = FALSE
)

```

### Arguments

<code>df</code>	A DataFrame of your data
<code>procs</code>	list of preprocess functions
<code>x_names</code>	predictors names
<code>y_names</code>	the names of the dependent variables
<code>block_y</code>	the TransformBlock to use for the target
<code>splits</code>	How to split your data
<code>do_setup</code>	A parameter for if Tabular will run the data through the procs upon initialization
<code>device</code>	device name
<code>inplace</code>	If True, Tabular will not keep a separate copy of your original DataFrame in memory

### Value

None

---

TabularTSDataloader    *TabularTSDataloader*

---

## Description

Transformed 'DataLoader'

## Usage

```
TabularTSDataloader(
    dataset,
    bs = 16,
    shuffle = FALSE,
    after_batch = NULL,
    num_workers = 0,
    verbose = FALSE,
    do_setup = TRUE,
    pin_memory = FALSE,
    timeout = 0,
    batch_size = NULL,
    drop_last = FALSE,
    indexed = NULL,
    n = NULL,
    device = NULL
)
```

## Arguments

dataset	data set
bs	batch size
shuffle	shuffle or not
after_batch	after batch
num_workers	the number of workers
verbose	verbose
do_setup	A parameter for if Tabular will run the data through the procs upon initialization
pin_memory	pin memory or not
timeout	timeout
batch_size	batch size
drop_last	drop last
indexed	indexed
n	n
device	device name

**Value**

None

---

tabular_config	<i>Tabular_config</i>
----------------	-----------------------

---

**Description**

Convenience function to easily create a config for ‘TabularModel‘

**Usage**

```
tabular_config(
  ps = NULL,
  embed_p = 0,
  y_range = NULL,
  use_bn = TRUE,
  bn_final = FALSE,
  bn_cont = TRUE,
  act_cls = nn()$ReLU(inplace = TRUE)
)
```

**Arguments**

ps	ps
embed_p	embed proportion
y_range	y_range
use_bn	use batch normalization
bn_final	batch normalization final
bn_cont	batch normalization
act_cls	activation

**Value**

None



---

tabular_learner	<i>Tabular learner</i>
-----------------	------------------------

---

### Description

Get a 'Learner' using 'dls', with 'metrics', including a 'TabularModel' created using the remaining params.

### Usage

```
tabular_learner(  
    dls,  
    layers = NULL,  
    emb_szs = NULL,  
    config = NULL,  
    n_out = NULL,  
    y_range = NULL,  
    loss_func = NULL,  
    opt_func = Adam(),  
    lr = 0.001,  
    splitter = trainable_params(),  
    cbs = NULL,  
    metrics = NULL,  
    path = NULL,  
    model_dir = "models",  
    wd = NULL,  
    wd_bn_bias = FALSE,  
    train_bn = TRUE,  
    moms = list(0.95, 0.85, 0.95)  
)
```

### Arguments

dls	It is a DataLoaders object.
layers	layers
emb_szs	emb_szs
config	config
n_out	n_out
y_range	y_range
loss_func	It can be any loss function you like.
opt_func	It will be used to create an optimizer when Learner.fit is called.
lr	It is learning rate.
splitter	It is a function that takes self.model and returns a list of parameter groups (or just one parameter group if there are no different parameter groups)

cbs	It is one or a list of Callbacks to pass to the Learner.
metrics	It is an optional list of metrics, that can be either functions or Metrics.
path	It is used to save and/or load models. Often path will be inferred from dls, but you can override it or pass a Path object to model_dir. Make sure you can write in path/model_dir!
model_dir	It is used to save and/or load models. Often path will be inferred from dls, but you can override it or pass a Path object to model_dir. Make sure you can write in path/model_dir!
wd	It is the default weight decay used when training the model.
wd_bn_bias	It controls if weight decay is applied to BatchNorm layers and bias.
train_bn	It controls if BatchNorm layers are trained even when they are supposed to be frozen according to the splitter.
moms	The default momentums used in Learner.fit_one_cycle.

**Value**

learner object

---

tar\_extract\_at\_filename  
*Tar\_extract\_at\_filename*

---

**Description**

Extract 'fname' to 'dest'/'fname.name' folder using 'tarfile'

**Usage**

```
tar_extract_at_filename(fname, dest)
```

**Arguments**

fname	folder name
dest	destination

**Value**

None

---

tensor	<i>Tensor</i>
--------	---------------

---

**Description**

Like ‘torch()\$as\_tensor’, but handle lists too, and can pass multiple vector elements directly.

**Usage**

```
tensor(...)
```

**Arguments**

... image

**Value**

None

---

TensorBBox	<i>TensorBBox</i>
------------	-------------------

---

**Description**

Basic type for a tensor of bounding boxes in an image

**Usage**

```
TensorBBox(x)
```

**Arguments**

x tensor

**Value**

None

TensorBBox\_create      *TensorBBox\_create*

---

**Description**

TensorBBox\_create

**Usage**

```
TensorBBox_create(x, img_size = NULL)
```

**Arguments**

x	tensor
img_size	image size

**Value**

None

---

TensorImage      *TensorImage*

---

**Description**

TensorImage

**Usage**

```
TensorImage(x)
```

**Arguments**

x	tensor
---	--------

**Value**

None

---

TensorImageBW

*TensorImageBW*

---

**Description**

TensorImageBW

**Usage**

TensorImageBW(x)

**Arguments**

x                    tensor

**Value**

None

---

TensorMultiCategory

*TensorMultiCategory*

---

**Description**

TensorMultiCategory

**Usage**

TensorMultiCategory(x)

**Arguments**

x                    tensor

**Value**

None

---

TensorPoint

*TensorPoint*

---

### Description

Basic type for points in an image

### Usage

TensorPoint(x)

### Arguments

x                    tensor

### Value

None

---

TensorPoint\_create

*TensorPoint\_create*

---

### Description

Delegates ('\_\_call\_\_', 'decode', 'setup') to ('encodes', 'decodes', 'setups') if 'split\_idx' matches

### Usage

TensorPoint\_create(...)

### Arguments

...                    arguments to pass

### Value

None

---

TerminateOnNaNCallback	<i>TerminateOnNaNCallback</i>
------------------------	-------------------------------

---

**Description**

TerminateOnNaNCallback

**Usage**

```
TerminateOnNaNCallback(...)
```

**Arguments**

... parameters to pass

**Value**

None

---

test_loader	<i>Test_loader</i>
-------------	--------------------

---

**Description**

Data loader. Combines a dataset and a sampler, and provides an iterable over

**Usage**

```
test_loader()
```

**Details**

the given dataset. The :class:`~torch.utils.data.DataLoader` supports both map-style and iterable-style datasets with single- or multi-process loading, customizing loading order and optional automatic batching (collation) and memory pinning. See :py:mod:`~torch.utils.data` documentation page for more details.

**Value**

loader

---

text	<i>Text module</i>
------	--------------------

---

**Description**

Text module

**Usage**

text()

**Value**

None

---

TextBlock	<i>TextBlock</i>
-----------	------------------

---

**Description**

A 'TransformBlock' for texts

**Usage**

```
TextBlock(
    tok_tfm,
    vocab = NULL,
    is_lm = FALSE,
    seq_len = 72,
    backwards = FALSE,
    min_freq = 3,
    max_vocab = 60000,
    special_toks = NULL,
    pad_tok = NULL
)
```

**Arguments**

tok_tfm	tok_tfm
vocab	vocab
is_lm	is_lm
seq_len	seq_len
backwards	backwards
min_freq	min_freq
max_vocab	max_vocab
special_toks	special_toks
pad_tok	pad_tok



**Value**

block object

---

TextBlock_from_df	<i>TextBlock_from_df</i>
-------------------	--------------------------

---

**Description**

Build a 'TextBlock' from a dataframe using 'text\_cols'

**Usage**

```
TextBlock_from_df(
    text_cols,
    vocab = NULL,
    is_lm = FALSE,
    seq_len = 72,
    backwards = FALSE,
    min_freq = 3,
    max_vocab = 60000,
    tok = NULL,
    rules = NULL,
    sep = " ",
    n_workers = 6,
    mark_fields = NULL,
    tok_text_col = "text"
)
```

**Arguments**

text_cols	text columns
vocab	vocabulary
is_lm	is_lm
seq_len	sequence length
backwards	backwards
min_freq	minimum frequency
max_vocab	max vocabulary
tok	tokenizer
rules	rules
sep	separator
n_workers	number workers
mark_fields	mark_fields
tok_text_col	result column name

**Value**

None

---

 TextBlock\_from\_folder *TextBlock\_from\_folder*


---

**Description**

Build a 'TextBlock' from a 'path'

**Usage**

```
TextBlock_from_folder(
    path,
    vocab = NULL,
    is_lm = FALSE,
    seq_len = 72,
    backwards = FALSE,
    min_freq = 3,
    max_vocab = 60000,
    tok = NULL,
    rules = NULL,
    extensions = NULL,
    folders = NULL,
    output_dir = NULL,
    skip_if_exists = TRUE,
    output_names = NULL,
    n_workers = 6,
    encoding = "utf8"
)
```

**Arguments**

path	path
vocab	vocabulary
is_lm	is_lm
seq_len	sequence length
backwards	backwards
min_freq	minimum frequency
max_vocab	max vocabulary
tok	tokenizer
rules	rules
extensions	extensions

folders	folders
output_dir	output_dir
skip_if_exists	skip_if_exists
output_names	output_names
n_workers	number of workers
encoding	encoding

**Value**

None

---

TextDataLoaders\_from\_csv

*TextDataLoaders\_from\_csv*


---

**Description**

Create from 'csv' file in 'path/csv\_fname'

**Usage**

```
TextDataLoaders_from_csv(
    path,
    csv_fname = "labels.csv",
    header = "infer",
    delimiter = NULL,
    valid_pct = 0.2,
    seed = NULL,
    text_col = 0,
    label_col = 1,
    label_delim = NULL,
    y_block = NULL,
    text_vocab = NULL,
    is_lm = FALSE,
    valid_col = NULL,
    tok_tfm = NULL,
    seq_len = 72,
    backwards = FALSE,
    bs = 64,
    val_bs = NULL,
    shuffle_train = TRUE,
    device = NULL
)
```

**Arguments**

path	path
csv_fname	csv file name
header	header
delimiter	delimiter
valid_pct	valid_ation percentage
seed	random seed
text_col	text column
label_col	label column
label_delim	label separator
y_block	y_block
text_vocab	text vocabulary
is_lm	is_lm
valid_col	valid column
tok_tfm	tok_tfm
seq_len	seq_len
backwards	backwards
bs	batch size
val_bs	validation batch size
shuffle_train	shuffle train data
device	device

**Value**

text loader

---

TextDataLoaders\_from\_df

*TextDataLoaders\_from\_df*

---

**Description**

Create from 'df' in 'path' with 'valid\_pct'

**Usage**

```
TextDataLoaders_from_df(
  df,
  path = ".",
  valid_pct = 0.2,
  seed = NULL,
  text_col = 0,
  label_col = 1,
  label_delim = NULL,
  y_block = NULL,
  text_vocab = NULL,
  is_lm = FALSE,
  valid_col = NULL,
  tok_tfm = NULL,
  seq_len = 72,
  backwards = FALSE,
  bs = 64,
  val_bs = NULL,
  shuffle_train = TRUE,
  device = NULL
)
```

**Arguments**

df	df
path	path
valid_pct	validation percentage
seed	seed
text_col	text_col
label_col	label_col
label_delim	label_delim
y_block	y_block
text_vocab	text_vocab
is_lm	is_lm
valid_col	valid_col
tok_tfm	tok_tfm
seq_len	seq_len
backwards	backwards
bs	batch size
val_bs	validation batch size, if not specified then val_bs is the same as bs.
shuffle_train	shuffle_train
device	device

**Value**

text loader

---

TextDataLoaders\_from\_folder

*TextDataLoaders\_from\_folder*


---

**Description**

Create from imagenet style dataset in ‘path‘ with ‘train‘ and ‘valid‘ subfolders (or provide ‘valid\_pct‘)

**Usage**

```
TextDataLoaders_from_folder(
    path,
    train = "train",
    valid = "valid",
    valid_pct = NULL,
    seed = NULL,
    vocab = NULL,
    text_vocab = NULL,
    is_lm = FALSE,
    tok_tfm = NULL,
    seq_len = 72,
    backwards = FALSE,
    bs = 64,
    val_bs = NULL,
    shuffle_train = TRUE,
    device = NULL
)
```

**Arguments**

path	path
train	train data
valid	validation data
valid_pct	validation percentage
seed	random seed
vocab	vocabulary
text_vocab	text_vocab
is_lm	is_lm
tok_tfm	tok_tfm
seq_len	seq_len

backwards	backwards
bs	batch size
val_bs	validation batch size
shuffle_train	shuffle train data
device	device

**Value**

text loader

---

TextLearner	<i>TextLearner</i>
-------------	--------------------

---

**Description**

Basic class for a ‘Learner’ in NLP.

**Usage**

```
TextLearner(
    dls,
    model,
    alpha = 2,
    beta = 1,
    moms = list(0.8, 0.7, 0.8),
    loss_func = NULL,
    opt_func = Adam(),
    lr = 0.001,
    splitter = trainable_params(),
    cbs = NULL,
    metrics = NULL,
    path = NULL,
    model_dir = "models",
    wd = NULL,
    wd_bn_bias = FALSE,
    train_bn = TRUE
)
```

**Arguments**

dls	dls
model	model
alpha	alpha
beta	beta
moms	moms

loss_func	loss_func
opt_func	opt_func
lr	lr
splitter	splitter
cbs	cbs
metrics	metrics
path	path
model_dir	model_dir
wd	wd
wd_bn_bias	wd_bn_bias
train_bn	train_bn

**Value**

None

---

TextLearner\_load\_encoder  
*Load\_encoder*

---

**Description**

Load the encoder 'file' from the model directory, optionally ensuring it's on 'device'

**Usage**

```
TextLearner_load_encoder(file, device = NULL)
```

**Arguments**

file	file
device	device

**Value**

None



---

TextLearner\_load\_pretrained  
*Load\_pretrained*

---

**Description**

Load a pretrained model and adapt it to the data vocabulary.

**Usage**

```
TextLearner_load_pretrained(wgts_fname, vocab_fname, model = NULL)
```

**Arguments**

wgts_fname	wgts_fname
vocab_fname	vocab_fname
model	model

**Value**

None

---

TextLearner\_save\_encoder  
*Save\_encoder*

---

**Description**

Save the encoder to 'file' in the model directory

**Usage**

```
TextLearner_save_encoder(file)
```

**Arguments**

file	file
------	------

**Value**

None

---

```
text_classifier_learner
    Text_classifier_learner
```

---

## Description

Create a 'Learner' with a text classifier from 'dls' and 'arch'.

## Usage

```
text_classifier_learner(  
    dls,  
    arch,  
    seq_len = 72,  
    config = NULL,  
    backwards = FALSE,  
    pretrained = TRUE,  
    drop_mult = 0.5,  
    n_out = NULL,  
    lin_ftrs = NULL,  
    ps = NULL,  
    max_len = 1440,  
    y_range = NULL,  
    loss_func = NULL,  
    opt_func = Adam(),  
    lr = 0.001,  
    splitter = trainable_params,  
    cbs = NULL,  
    metrics = NULL,  
    path = NULL,  
    model_dir = "models",  
    wd = NULL,  
    wd_bn_bias = FALSE,  
    train_bn = TRUE,  
    moms = list(0.95, 0.85, 0.95)  
)
```

## Arguments

dls	dls
arch	arch
seq_len	seq_len
config	config
backwards	backwards
pretrained	pretrained

drop_mult	drop_mult
n_out	n_out
lin_ftrs	lin_ftrs
ps	ps
max_len	max_len
y_range	y_range
loss_func	loss_func
opt_func	opt_func
lr	lr
splitter	splitter
cbs	cbs
metrics	metrics
path	path
model_dir	model_dir
wd	wd
wd_bn_bias	wd_bn_bias
train_bn	train_bn
moms	moms

**Value**

None

---

TfmdDL

*TfmdDL*


---

**Description**

Transformed 'DataLoader'

**Usage**

```
TfmdDL(
    dataset,
    bs = 64,
    shuffle = FALSE,
    num_workers = NULL,
    verbose = FALSE,
    do_setup = TRUE,
    pin_memory = FALSE,
    timeout = 0,
    batch_size = NULL,
```

```

    drop_last = FALSE,
    indexed = NULL,
    n = NULL,
    device = NULL,
    after_batch = NULL,
    ...
)

```

### Arguments

dataset	dataset
bs	batch size
shuffle	shuffle
num_workers	number of workers
verbose	verbose
do_setup	do setup
pin_memory	pin memory
timeout	timeout
batch_size	batch size
drop_last	drop last
indexed	indexed
n	int, n
device	device
after_batch	after_batch
...	additional arguments to pass

### Value

None

---

TfmdLists

*TfmdLists*

---

### Description

A ‘Pipeline’ of ‘tfms’ applied to a collection of ‘items’

### Usage

```
TfmdLists(...)
```

### Arguments

... parameters to pass

---

TfmResize	<i>TfmResize</i>
-----------	------------------

---

**Description**

Temporary fix to allow image resizing transform

**Usage**

```
TfmResize(size, interp_mode = "bilinear")
```

**Arguments**

size	size
interp_mode	interpolation mode

**Value**

None

---

timm	<i>Timm module</i>
------	--------------------

---

**Description**

Timm module

**Usage**

```
timm()
```

**Value**

None

---

timm_learner	<i>Timm_learner</i>
--------------	---------------------

---

**Description**

Build a convnet style learner from 'dls' and 'arch' using the 'timm' library

**Usage**

```
timm_learner(dls, arch, ...)
```

**Arguments**

dls	dataloader
arch	model architecture
...	additional arguments

**Value**

None

---

timm_list_models	<i>Timm models</i>
------------------	--------------------

---

**Description**

Timm models

**Usage**

```
timm_list_models(...)
```

**Arguments**

...	parameters to pass
-----	--------------------

**Value**

vector

---

tms	<i>Timeseries module</i>
-----	--------------------------

---

**Description**

Timeseries module

**Usage**

tms()

**Value**

None

---

tokenize1	<i>Tokenize1</i>
-----------	------------------

---

**Description**

Call 'TokenizeWithRules' with a single text

**Usage**

tokenize1(text, tok, rules = NULL, post\_rules = NULL)

**Arguments**

text	text
tok	tok
rules	rules
post_rules	post_rules

**Value**

None

---

 Tokenizer

*Tokenizer*


---

**Description**

Provides a consistent ‘Transform’ interface to tokenizers operating on ‘DataFrame’s and folders

**Usage**

```
Tokenizer(
  tok,
  rules = NULL,
  counter = NULL,
  lengths = NULL,
  mode = NULL,
  sep = " "
)
```

**Arguments**

tok	tokenizer
rules	rules
counter	counter
lengths	lengths
mode	mode
sep	separator

**Value**

None

---

 Tokenizer\_from\_df

*Tokenizer\_from\_df*


---

**Description**

Tokenizer\_from\_df



**Usage**

```

Tokenizer_from_df(
  text_cols,
  tok = NULL,
  rules = NULL,
  sep = " ",
  n_workers = 6,
  mark_fields = NULL,
  tok_text_col = "text"
)

```

**Arguments**

text_cols	text columns
tok	tokenizer
rules	special rules
sep	separator
n_workers	number of workers
mark_fields	mark fields
tok_text_col	output column name

**Value**

None

---

TokenizeWithRules	<i>TokenizeWithRules</i>
-------------------	--------------------------

---

**Description**

A wrapper around ‘tok’ which applies ‘rules’, then tokenizes, then applies ‘post\_rules’

**Usage**

```
TokenizeWithRules(tok, rules = NULL, post_rules = NULL)
```

**Arguments**

tok	tokenizer
rules	rules
post_rules	post_rules

**Value**

None

---

tokenize_csv	<i>Tokenize_csv</i>
--------------	---------------------

---

### Description

Tokenize texts in the 'text\_cols' of the csv 'fname' in parallel using 'n\_workers'

### Usage

```
tokenize_csv(  
    fname,  
    text_cols,  
    outname = NULL,  
    n_workers = 4,  
    rules = NULL,  
    mark_fields = NULL,  
    tok = NULL,  
    header = "infer",  
    chunksize = 50000  
)
```

### Arguments

fname	file name
text_cols	text columns
outname	outname
n_workers	number of workers
rules	rules
mark_fields	mark fields
tok	tokenizer
header	header
chunksize	chunk size

### Value

None

---

tokenize_df	<i>Tokenize_df</i>
-------------	--------------------

---

### Description

Tokenize texts in 'df[text\_cols]' in parallel using 'n\_workers'

### Usage

```
tokenize_df(
  df,
  text_cols,
  n_workers = 6,
  rules = NULL,
  mark_fields = NULL,
  tok = NULL,
  tok_text_col = "text"
)
```

### Arguments

df	data frame
text_cols	text columns
n_workers	number of workers
rules	rules
mark_fields	mark_fields
tok	tokenizer
tok_text_col	tok_text_col

### Value

None

---

tokenize_files	<i>Tokenize_files</i>
----------------	-----------------------

---

### Description

Tokenize text 'files' in parallel using 'n\_workers'

**Usage**

```

tokenize_files(
  files,
  path,
  output_dir,
  output_names = NULL,
  n_workers = 6,
  rules = NULL,
  tok = NULL,
  encoding = "utf8",
  skip_if_exists = FALSE
)

```

**Arguments**

files	files
path	path
output_dir	output_dir
output_names	output_names
n_workers	n_workers
rules	rules
tok	tokenizer
encoding	encoding
skip_if_exists	skip_if_exists

**Value**

None

---

tokenize_folder	<i>Tokenize_folder</i>
-----------------	------------------------

---

**Description**

Tokenize text files in ‘path‘ in parallel using ‘n\_workers‘

**Usage**

```

tokenize_folder(
  path,
  extensions = NULL,
  folders = NULL,
  output_dir = NULL,
  skip_if_exists = TRUE,
)

```

```

    output_names = NULL,
    n_workers = 6,
    rules = NULL,
    tok = NULL,
    encoding = "utf8"
)

```

### Arguments

path	path
extensions	extensions
folders	folders
output_dir	output_dir
skip_if_exists	skip_if_exists
output_names	output_names
n_workers	number of workers
rules	rules
tok	tokenizer
encoding	encoding

### Value

None

---

tokenize_texts	<i>Tokenize_texts</i>
----------------	-----------------------

---

### Description

Tokenize ‘texts’ in parallel using ‘n\_workers’

### Usage

```
tokenize_texts(texts, n_workers = 6, rules = NULL, tok = NULL)
```

### Arguments

texts	texts
n_workers	n_workers
rules	rules
tok	tok

### Value

None

---

top_k_accuracy	<i>Top_k_accuracy</i>
----------------	-----------------------

---

**Description**

Computes the Top-k accuracy ('targ' is in the top 'k' predictions of 'inp')

**Usage**

```
top_k_accuracy(inp, targ, k = 5, axis = -1)
```

**Arguments**

inp	predictions
targ	targets
k	k
axis	axis

**Value**

None

**Examples**

```
## Not run:  
  
loaders = loaders()  
  
data = Data_Loaders(loaders['train'], loaders['valid'])$cuda()  
  
model = nn$Sequential() +  
  nn$Flatten() +  
  nn$Linear(28L * 28L, 10L)  
metrics = list(accuracy, top_k_accuracy)  
learn = Learner(data, model, loss_func = F$cross_entropy, opt_func = Adam,  
  metrics = metrics)  
  
## End(Not run)
```

---

torch	<i>Builtins module</i>
-------	------------------------

---

**Description**

Builtins module

**Usage**

torch()

**Value**

None

---

total_params	<i>Total_params</i>
--------------	---------------------

---

**Description**

Give the number of parameters of a module and if it's trainable or not

**Usage**

total\_params(m)

**Arguments**

m            m parameter

**Value**

None

---

ToTensor	<i>ToTensor</i>
----------	-----------------

---

**Description**

Convert item to appropriate tensor class

**Usage**

```
ToTensor(enc = NULL, dec = NULL, split_idx = NULL, order = NULL)
```

**Arguments**

enc	encoder
dec	decoder
split_idx	int, split by index
order	order

**Value**

None

---

to_bytes_format	<i>To_bytes_format</i>
-----------------	------------------------

---

**Description**

Convert to bytes, default to PNG format

**Usage**

```
to_bytes_format(img, format = "png")
```

**Arguments**

img	image
format	format

**Value**

None



---

to_image	<i>To_image</i>
----------	-----------------

---

**Description**

Convert a tensor or array to a PIL int8 Image

**Usage**

```
to_image(x)
```

**Arguments**

x	tensor
---	--------

**Value**

None

---

to_matrix	<i>To_matrix</i>
-----------	------------------

---

**Description**

To matrix

**Usage**

```
to_matrix(obj, matrix = TRUE)
```

**Arguments**

obj	learner/model
matrix	bool, to R matrix

---

to_thumb	<i>To_thumb</i>
----------	-----------------

---

**Description**

Same as 'thumbnail', but uses a copy

**Usage**

```
to_thumb(img, h, w = NULL)
```

**Arguments**

img	image
h	height
w	width

**Value**

None

---

to_xla	<i>Learn to XLA</i>
--------	---------------------

---

**Description**

Distribute the training across TPUs

**Usage**

```
to_xla(object)
```

**Arguments**

object	learner / model
--------	-----------------

**Value**

None

---

TrackerCallback	<i>TrackerCallback</i>
-----------------	------------------------

---

**Description**

A 'Callback' that keeps track of the best value in 'monitor'.

**Usage**

```
TrackerCallback(monitor = "valid_loss", comp = NULL, min_delta = 0)
```

**Arguments**

monitor	monitor the loss
comp	comp
min_delta	minimum delta

**Value**

None

---

trainable_params	<i>Trainable_params</i>
------------------	-------------------------

---

**Description**

Return all trainable parameters of 'm'

**Usage**

```
trainable_params(m)
```

**Arguments**

m	trainable parameters
---	----------------------

**Value**

None

---

TrainEvalCallback	<i>TrainEvalCallback</i>
-------------------	--------------------------

---

**Description**

TrainEvalCallback

**Usage**

```
TrainEvalCallback(...)
```

**Arguments**

... parameters to pass

**Value**

None

---

train_loader	<i>Train_loader</i>
--------------	---------------------

---

**Description**

Data loader. Combines a dataset and a sampler, and provides an iterable over

**Usage**

```
train_loader()
```

**Details**

the given dataset. The :class:`~torch.utils.data.DataLoader` supports both map-style and iterable-style datasets with single- or multi-process loading, customizing loading order and optional automatic batching (collation) and memory pinning.

**Value**

loader

---

Transform	<i>Transform</i>
-----------	------------------

---

**Description**

Delegates (`__call__`, `decode`, `setup`) to (`encodes`, `decodes`, `setups`) if `split_idx` matches

**Usage**

```
Transform(enc = NULL, dec = NULL, split_idx = NULL, order = NULL)
```

**Arguments**

<code>enc</code>	encoder
<code>dec</code>	decoder
<code>split_idx</code>	split by index
<code>order</code>	order

**Value**

None

---

TransformBlock	<i>TransformBlock</i>
----------------	-----------------------

---

**Description**

A basic wrapper that links defaults transforms for the data block API

**Usage**

```
TransformBlock(
    type_tfms = NULL,
    item_tfms = NULL,
    batch_tfms = NULL,
    dl_type = NULL,
    dls_kwargs = NULL
)
```

**Arguments**

<code>type_tfms</code>	transformation type
<code>item_tfms</code>	item transformation type
<code>batch_tfms</code>	one or several transforms applied to the batches once they are formed
<code>dl_type</code>	DL application
<code>dls_kwargs</code>	additional arguments

**Value**

block

---

transformers

*Transformers*

---

**Description**

Transformers

**Usage**

transformers()

**Value**

None

---

TransformersDropOutput

*TransformersDropOutput*

---

**Description**

TransformersDropOutput

**Usage**

TransformersDropOutput()

**Value**

None

---

TransformersTokenizer *TransformersTokenizer*

---

**Description**

TransformersTokenizer

**Usage**

TransformersTokenizer(tokenizer)

**Arguments**

tokenizer      tokenizer object

**Value**

None

---

trunc\_normal\_      *Trunc\_normal\_*

---

**Description**

Truncated normal initialization (approximation)

**Usage**

trunc\_normal\_(x, mean = 0, std = 1)

**Arguments**

x                  tensor  
mean                mean  
std                 standard deviation

**Value**

tensor

---

TSBlock	<i>TSBlock</i>
---------	----------------

---

**Description**

A TimeSeries Block to process one timeseries

**Usage**

```
TSBlock(...)
```

**Arguments**

... parameters to pass

**Value**

None

---

TSDataLoaders_from_dfs	<i>TSDataLoaders_from_dfs</i>
------------------------	-------------------------------

---

**Description**

Create a DataLoader from a df\_train and df\_valid

**Usage**

```
TSDataLoaders_from_dfs(
    df_train,
    df_valid,
    path = ".",
    x_cols = NULL,
    label_col = NULL,
    y_block = NULL,
    item_tfms = NULL,
    batch_tfms = NULL,
    bs = 64,
    val_bs = NULL,
    shuffle_train = TRUE,
    device = NULL
)
```



**Arguments**

df_train	train data
df_valid	validation data
path	path (optional)
x_cols	predictors
label_col	label/output column
y_block	y_block
item_tfms	item transformations
batch_tfms	batch transformations
bs	batch size
val_bs	validation batch size
shuffle_train	shuffle train data
device	device name

**Value**

None

---

TSDDataTable

*TSDDataTable*


---

**Description**

A ‘DataFrame‘ wrapper that knows which cols are x/y, and returns rows in ‘\_\_getitem\_\_‘

**Usage**

```
TSDDataTable(
    df,
    procs = NULL,
    x_names = NULL,
    y_names = NULL,
    block_y = NULL,
    splits = NULL,
    do_setup = TRUE,
    device = NULL,
    inplace = FALSE
)
```

**Arguments**

df	A DataFrame of your data
procs	list of preprocess functions
x_names	predictors names
y_names	the names of the dependent variables
block_y	the TransformBlock to use for the target
splits	How to split your data
do_setup	A parameter for if Tabular will run the data through the procs upon initialization
device	device name
inplace	If True, Tabular will not keep a separate copy of your original DataFrame in memory

**Value**

None

---

TSeries

*TSeries*


---

**Description**

Basic Time series wrapper

**Usage**

TSeries(...)

**Arguments**

... parameters to pass

**Value**

None

---

TSeries_create	<i>TSeries_create</i>
----------------	-----------------------

---

**Description**

TSeries\_create

**Usage**

```
TSeries_create(x, ...)
```

**Arguments**

x	tensor
...	additional parameters

**Value**

tensor

**Examples**

```
## Not run:  
  
res = TSeries_create(as.array(runif(100)))  
res %>% show(title = 'R array') %>% plot(dpi = 200)  
  
## End(Not run)
```

---

UnetBlock	<i>UnetBlock</i>
-----------	------------------

---

**Description**

A quasi-UNet block, using 'PixelShuffle\_ICNR upsampling'.

**Usage**

```

UnetBlock(
  up_in_c,
  x_in_c,
  hook,
  final_div = TRUE,
  blur = FALSE,
  act_cls = nn()$ReLU,
  self_attention = FALSE,
  init = nn()$init$kaiming_normal_,
  norm_type = NULL,
  ks = 3,
  stride = 1,
  padding = NULL,
  bias = NULL,
  ndim = 2,
  bn_1st = TRUE,
  transpose = FALSE,
  xtra = NULL,
  bias_std = 0.01,
  dilation = 1,
  groups = 1,
  padding_mode = "zeros"
)

```

**Arguments**

up_in_c	up_in_c parameter
x_in_c	x_in_c parameter
hook	The hook is set to this intermediate layer to store the output needed for this block.
final_div	final div
blur	blur is used to avoid checkerboard artifacts at each layer.
act_cls	activation
self_attention	self_attention determines if we use a self-attention layer
init	initializer
norm_type	normalization type
ks	kernel size
stride	stride
padding	padding mode
bias	bias
ndim	number of dimensions
bn_1st	batch normalization 1st
transpose	transpose

extra	extra
bias_std	bias standard deviation
dilation	dilation
groups	groups
padding_mode	The mode of padding

**Value**

None

---

UNET_CONFIG	<i>UNET_CONFIG</i>
-------------	--------------------

---

**Description**

Convenience function to easily create a config for 'DynamicUNET'

**Usage**

```
UNET_CONFIG(
  blur = FALSE,
  blur_final = TRUE,
  self_attention = FALSE,
  y_range = NULL,
  last_cross = TRUE,
  bottle = FALSE,
  act_cls = nn()$ReLU,
  init = nn()$init$kaiming_normal_,
  norm_type = NULL
)
```

**Arguments**

blur	blur is used to avoid checkerboard artifacts at each layer.
blur_final	blur final is specific to the last layer.
self_attention	self_attention determines if we use a self attention layer at the third block before the end.
y_range	If y_range is passed, the last activations go through a sigmoid rescaled to that range.
last_cross	last cross
bottle	bottle
act_cls	activation
init	initializer
norm_type	normalization type

**Value**

None

---

UNET_learner	<i>UNET_learner</i>
--------------	---------------------

---

**Description**

Build a UNET learner from 'dls' and 'arch'

**Usage**

UNET\_learner(dls, arch, ...)

**Arguments**

dls	dataloader
arch	architecture
...	additional arguments

**Value**

None

---

unfreeze	<i>Unfreeze a model</i>
----------	-------------------------

---

**Description**

Unfreeze a model

**Usage**

unfreeze(object, ...)

**Arguments**

object	A model
...	Additional parameters

**Value**

None

**Examples**

```
## Not run:  
learnR %>% unfreeze()  
  
## End(Not run)
```

---

uniform_blur2d	<i>Uniform_blur2d</i>
----------------	-----------------------

---

**Description**

Uniformly apply blurring

**Usage**

```
uniform_blur2d(x, s)
```

**Arguments**

x	image
s	effect

**Value**

None

---

upit	<i>Upit module</i>
------	--------------------

---

**Description**

Upit module

**Usage**

```
upit()
```

**Value**

None

URLs\_ADULT\_SAMPLE      *ADULT\_SAMPLE dataset*

---

**Description**

download ADULT\_SAMPLE dataset

**Usage**

```
URLs_ADULT_SAMPLE(filename = "ADULT_SAMPLE", untar = TRUE)
```

**Arguments**

filename      the name of the file  
untar          logical, whether to untar the '.tgz' file

**Value**

None

**Examples**

```
## Not run:  
  
URLs_ADULT_SAMPLE()  
  
## End(Not run)
```

---

URLs\_AG\_NEWS          *AG\_NEWS dataset*

---

**Description**

download AG\_NEWS dataset

**Usage**

```
URLs_AG_NEWS(filename = "AG_NEWS", untar = TRUE)
```

**Arguments**

filename      the name of the file  
untar          logical, whether to untar the '.tgz' file



**Value**

None

**Examples**

```
## Not run:
```

```
URLs_AG_NEWS()
```

```
## End(Not run)
```

---

```
URLs_AMAZON_REVIEWSAMAZON_REVIEWS
```

```
AMAZON_REVIEWSAMAZON_REVIEWS dataset
```

---

**Description**

download AMAZON\_REVIEWSAMAZON\_REVIEWS dataset

**Usage**

```
URLs_AMAZON_REVIEWSAMAZON_REVIEWS(  
  filename = "AMAZON_REVIEWSAMAZON_REVIEWS",  
  untar = TRUE  
)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_AMAZON\_REVIEWS\_POLARITY  
*AMAZON\_REVIEWS\_POLARITY dataset*

---

**Description**

download AMAZON\_REVIEWS\_POLARITY dataset

**Usage**

```
URLs_AMAZON_REVIEWS_POLARITY(  
  filename = "AMAZON_REVIEWS_POLARITY",  
  untar = TRUE  
)
```

**Arguments**

filename        the name of the file  
untar           logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_BIWI\_HEAD\_POSE    *BIWI\_HEAD\_POSE dataset*

---

**Description**

download BIWI\_HEAD\_POSE dataset

**Usage**

```
URLs_BIWI_HEAD_POSE(filename = "BIWI_HEAD_POSE", untar = TRUE)
```

**Arguments**

filename        the name of the file  
untar           logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_CALTECH_101	<i>CALTECH_101 dataset</i>
------------------	----------------------------

---

**Description**

download CALTECH\_101 dataset

**Usage**

```
URLs_CALTECH_101(filename = "CALTECH_101", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_CAMVID	<i>CAMVID dataset</i>
-------------	-----------------------

---

**Description**

download CAMVID dataset

**Usage**

```
URLs_CAMVID(filename = "CAMVID", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_CAMVID_TINY	<i>CAMVID_TINY dataset</i>
------------------	----------------------------

---

**Description**

download CAMVID\_TINY dataset

**Usage**

```
URLs_CAMVID_TINY(filename = "CAMVID_TINY", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_CARS	<i>CARS dataset</i>
-----------	---------------------

---

**Description**

download CARS dataset

**Usage**

```
URLs_CARS(filename = "CARS", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_CIFAR	<i>CIFAR dataset</i>
------------	----------------------

---

**Description**

download CIFAR dataset

**Usage**

```
URLs_CIFAR(filename = "CIFAR", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_CIFAR_100	<i>CIFAR_100 dataset</i>
----------------	--------------------------

---

**Description**

download CIFAR\_100 dataset

**Usage**

```
URLs_CIFAR_100(filename = "CIFAR_100", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_COCO\_TINY      *COCO\_TINY dataset*

---

**Description**

download COCO\_TINY dataset

**Usage**

```
URLs_COCO_TINY(filename = "COCO_TINY", untar = TRUE)
```

**Arguments**

filename      the name of the file  
untar          logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_CUB\_200\_2011      *CUB\_200\_2011 dataset*

---

**Description**

download CUB\_200\_2011 dataset

**Usage**

```
URLs_CUB_200_2011(filename = "CUB_200_2011", untar = TRUE)
```

**Arguments**

filename      the name of the file  
untar          logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_DBPEDIA	<i>DBPEDIA dataset</i>
--------------	------------------------

---

**Description**

download DBPEDIA dataset

**Usage**

```
URLs_DBPEDIA(filename = "DBPEDIA", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_DOGS	<i>DOGS dataset</i>
-----------	---------------------

---

**Description**

download DOGS dataset

**Usage**

```
URLs_DOGS(filename = "DOGS", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_FLOWERS	<i>FLOWERS dataset</i>
--------------	------------------------

---

**Description**

download FLOWERS dataset

**Usage**

```
URLs_FLOWERS(filename = "FLOWERS", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_FOOD	<i>FOOD dataset</i>
-----------	---------------------

---

**Description**

download FOOD dataset

**Usage**

```
URLs_FOOD(filename = "FOOD", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None



---

URLs\_HORSE\_2\_ZEBRA     *HORSE\_2\_ZEBRA dataset*

---

**Description**

download HORSE\_2\_ZEBRA dataset

**Usage**

```
URLs_HORSE_2_ZEBRA(filename = "horse2zebra", unzip = TRUE)
```

**Arguments**

filename	the name of the file
unzip	logical, whether to unzip the '.zip' file

**Value**

None

---

URLs\_HUMAN\_NUMBERS     *HUMAN\_NUMBERS dataset*

---

**Description**

download HUMAN\_NUMBERS dataset

**Usage**

```
URLs_HUMAN_NUMBERS(filename = "HUMAN_NUMBERS", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_IMAGENETTE	<i>IMAGENETTE dataset</i>
-----------------	---------------------------

---

**Description**

download IMAGENETTE dataset

**Usage**

```
URLs_IMAGENETTE(filename = "IMAGENETTE", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_IMAGENETTE_160	<i>IMAGENETTE_160 dataset</i>
---------------------	-------------------------------

---

**Description**

download IMAGENETTE\_160 dataset

**Usage**

```
URLs_IMAGENETTE_160(filename = "IMAGENETTE_160", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_IMAGENETTE\_320     *IMAGENETTE\_320 dataset*

---

**Description**

download IMAGENETTE\_320 dataset

**Usage**

```
URLs_IMAGENETTE_320(filename = "IMAGENETTE_320", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_IMAGEWOOF     *IMAGEWOOF dataset*

---

**Description**

download IMAGEWOOF dataset

**Usage**

```
URLs_IMAGEWOOF(filename = "IMAGEWOOF", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_IMAGEWOOF\_160     *IMAGEWOOF\_160 dataset*

---

**Description**

download IMAGEWOOF\_160 dataset

**Usage**

```
URLs_IMAGEWOOF_160(filename = "IMAGEWOOF_160", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_IMAGEWOOF\_320     *IMAGEWOOF\_320 dataset*

---

**Description**

download IMAGEWOOF\_320 dataset

**Usage**

```
URLs_IMAGEWOOF_320(filename = "IMAGEWOOF_320", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_IMDB	<i>IMDB dataset</i>
-----------	---------------------

---

**Description**

download IMDB dataset

**Usage**

```
URLs_IMDB(filename = "IMDB", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_IMDB_SAMPLE	<i>IMDB_SAMPLE dataset</i>
------------------	----------------------------

---

**Description**

download IMDB\_SAMPLE dataset

**Usage**

```
URLs_IMDB_SAMPLE(filename = "IMDB_SAMPLE", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

URLs\_LSUN\_BEDROOMS      *LSUN\_BEDROOMS dataset*

---

**Description**

download LSUN\_BEDROOMS dataset

**Usage**

```
URLs_LSUN_BEDROOMS(filename = "LSUN_BEDROOMS", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_ML\_SAMPLE      *ML\_SAMPLE dataset*

---

**Description**

download ML\_SAMPLE dataset

**Usage**

```
URLs_ML_SAMPLE(filename = "ML_SAMPLE", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_MNIST	<i>MNIST dataset</i>
------------	----------------------

---

**Description**

download MNIST dataset

**Usage**

```
URLs_MNIST(filename = "MNIST", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_MNIST_SAMPLE	<i>MNIST_SAMPLE dataset</i>
-------------------	-----------------------------

---

**Description**

download MNIST\_SAMPLE dataset

**Usage**

```
URLs_MNIST_SAMPLE(filename = "MNIST_SAMPLE", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_MNIST\_TINY      *MNIST\_TINY dataset*

---

**Description**

download MNIST\_TINY dataset

**Usage**

```
URLs_MNIST_TINY(filename = "MNIST_TINY", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_MNIST\_VAR\_SIZE\_TINY  
*MNIST\_VAR\_SIZE\_TINY dataset*

---

**Description**

download MNIST\_VAR\_SIZE\_TINY dataset

**Usage**

```
URLs_MNIST_VAR_SIZE_TINY(filename = "MNIST_VAR_SIZE_TINY", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None



---

URLs\_MOVIE\_LENS\_ML\_100k  
*MOVIE\_LENS\_ML\_100k dataset*

---

**Description**

download MOVIE\_LENS\_ML\_100k dataset

**Usage**

```
URLs_MOVIE_LENS_ML_100k(filename = "ml-100k", unzip = TRUE)
```

**Arguments**

filename	the name of the file
unzip	logical, whether to unzip the '.zip' file

**Value**

None

---

URLs\_MT\_ENG\_FRA      *MT\_ENG\_FRA dataset*

---

**Description**

download MT\_ENG\_FRA dataset

**Usage**

```
URLs_MT_ENG_FRA(filename = "MT_ENG_FRA", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_OPENAI\_TRANSFORMER

*OPENAI\_TRANSFORMER dataset*

---

**Description**

download OPENAI\_TRANSFORMER dataset

**Usage**

```
URLs_OPENAI_TRANSFORMER(filename = "OPENAI_TRANSFORMER", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_PASCAL\_2007

*PASCAL\_2007 dataset*

---

**Description**

download PASCAL\_2007 dataset

**Usage**

```
URLs_PASCAL_2007(filename = "PASCAL_2007", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_PASCAL_2012	<i>PASCAL_2012 dataset</i>
------------------	----------------------------

---

**Description**

download PASCAL\_2012 dataset

**Usage**

```
URLs_PASCAL_2012(filename = "PASCAL_2012", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_PETS	<i>PETS dataset</i>
-----------	---------------------

---

**Description**

download PETS dataset

**Usage**

```
URLs_PETS(filename = "PETS", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

URLs\_PLANET\_SAMPLE     *PLANET\_SAMPLE dataset*

---

**Description**

download PLANET\_SAMPLE dataset

**Usage**

```
URLs_PLANET_SAMPLE(filename = "PLANET_SAMPLE", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_PLANET\_TINY     *PLANET\_TINY dataset*

---

**Description**

download PLANET\_TINY dataset

**Usage**

```
URLs_PLANET_TINY(filename = "PLANET_TINY", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_S3_COCO	<i>S3_COCO dataset</i>
--------------	------------------------

---

**Description**

download S3\_COCO dataset

**Usage**

```
URLs_S3_COCO(filename = "S3_COCO", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_S3_IMAGE	<i>S3_IMAGE dataset</i>
---------------	-------------------------

---

**Description**

download S3\_IMAGE dataset

**Usage**

```
URLs_S3_IMAGE(filename = "S3_IMAGE", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

URLs\_S3\_IMAGELOC      *S3\_IMAGELOC dataset*

---

**Description**

download S3\_IMAGELOC dataset

**Usage**

```
URLs_S3_IMAGELOC(filename = "S3_IMAGELOC", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_S3\_MODEL      *S3\_MODEL dataset*

---

**Description**

download S3\_MODEL dataset

**Usage**

```
URLs_S3_MODEL(filename = "S3_MODEL", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_S3_NLP	<i>S3_NLP dataset</i>
-------------	-----------------------

---

**Description**

download S3\_NLP dataset

**Usage**

```
URLs_S3_NLP(filename = "S3_NLP", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_SIIM_SMALL	<i>SIIM_SMALL</i>
-----------------	-------------------

---

**Description**

download YELP\_REVIEWS\_POLARITY dataset

**Usage**

```
URLs_SIIM_SMALL(filename = "SIIM_SMALL", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_SKIN\_LESION      *SKIN\_LESION dataset*

---

**Description**

download SKIN\_LESION dataset

**Usage**

```
URLs_SKIN_LESION(filename = "SKIN_LESION", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_SOGOU\_NEWS      *SOGOU\_NEWS dataset*

---

**Description**

download SOGOU\_NEWS dataset

**Usage**

```
URLs_SOGOU_NEWS(filename = "SOGOU_NEWS", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None



---

URLs_SPEAKERS10	<i>SPEAKERS10 dataset</i>
-----------------	---------------------------

---

**Description**

download SPEAKERS10 dataset

**Usage**

```
URLs_SPEAKERS10(filename = "SPEAKERS10", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

**Examples**

```
## Not run:  
  
URLs_SPEAKERS10()  
  
## End(Not run)
```

---

URLs_SPEECHCOMMANDS	<i>SPEECHCOMMANDS dataset</i>
---------------------	-------------------------------

---

**Description**

download SPEECHCOMMANDS dataset

**Usage**

```
URLs_SPEECHCOMMANDS(filename = "SPEECHCOMMANDS", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

**Examples**

```
## Not run:  
  
URLs_SPEECHCOMMANDS()  
  
## End(Not run)
```

---

URLs_WIKITEXT	<i>WIKITEXT dataset</i>
---------------	-------------------------

---

**Description**

download WIKITEXT dataset

**Usage**

```
URLs_WIKITEXT(filename = "WIKITEXT", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_WIKITEXT_TINY	<i>WIKITEXT_TINY dataset</i>
--------------------	------------------------------

---

**Description**

download WIKITEXT\_TINY dataset

**Usage**

```
URLs_WIKITEXT_TINY(filename = "WIKITEXT_TINY", untar = TRUE)
```

**Arguments**

filename        the name of the file  
 untar           logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_WT103_BWD	<i>WT103_BWD dataset</i>
----------------	--------------------------

---

**Description**

download WT103\_BWD dataset

**Usage**

```
URLs_WT103_BWD(filename = "WT103_BWD", untar = TRUE)
```

**Arguments**

filename        the name of the file  
 untar           logical, whether to untar the '.tgz' file

**Value**

None

---

URLs_WT103_FWD	<i>WT103_FWD dataset</i>
----------------	--------------------------

---

**Description**

download WT103\_FWD dataset

**Usage**

```
URLs_WT103_FWD(filename = "WT103_FWD", untar = TRUE)
```

**Arguments**

filename        the name of the file  
 untar           logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_YAHOO\_ANSWERS      *YAHOO\_ANSWERS dataset*

---

**Description**

download YAHOO\_ANSWERS dataset

**Usage**

```
URLs_YAHOO_ANSWERS(filename = "YAHOO_ANSWERS", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_YELP\_REVIEWS      *YELP\_REVIEWS dataset*

---

**Description**

download YELP\_REVIEWS dataset

**Usage**

```
URLs_YELP_REVIEWS(filename = "YELP_REVIEWS", untar = TRUE)
```

**Arguments**

filename	the name of the file
untar	logical, whether to untar the '.tgz' file

**Value**

None

---

URLs\_YELP\_REVIEWS\_POLARITY  
*YELP\_REVIEWS\_POLARITY dataset*

---

**Description**

download YELP\_REVIEWS\_POLARITY dataset

**Usage**

URLs\_YELP\_REVIEWS\_POLARITY(filename = "YELP\_REVIEWS\_POLARITY", untar = TRUE)

**Arguments**

filename        the name of the file  
 untar            logical, whether to untar the '.tgz' file

**Value**

None

---

vgg11\_bn                *Vgg11\_bn*

---

**Description**

VGG 11-layer model (configuration "A") with batch normalization

**Usage**

vgg11\_bn(pretrained = FALSE, progress)

**Arguments**

pretrained        pretrained or not  
 progress          to see progress bar or not

**Details**

"Very Deep Convolutional Networks For Large-Scale Image Recognition" <<https://arxiv.org/pdf/1409.1556.pdf>>

**Value**

model

---

 vgg13\_bn

*Vgg13\_bn*


---

**Description**

VGG 13-layer model (configuration "B") with batch normalization

**Usage**

```
vgg13_bn(pretrained = FALSE, progress)
```

**Arguments**

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

"Very Deep Convolutional Networks For Large-Scale Image Recognition" <<https://arxiv.org/pdf/1409.1556.pdf>>

**Value**

model

---

vgg16\_bn

*Vgg16\_bn*


---

**Description**

VGG 16-layer model (configuration "D") with batch normalization

**Usage**

```
vgg16_bn(pretrained = FALSE, progress)
```

**Arguments**

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

"Very Deep Convolutional Networks For Large-Scale Image Recognition" <<https://arxiv.org/pdf/1409.1556.pdf>>

**Value**

model

---

 vgg19\_bn

*Vgg19\_bn*


---

**Description**

VGG 19-layer model (configuration 'E') with batch normalization

**Usage**

```
vgg19_bn(pretrained = FALSE, progress)
```

**Arguments**

pretrained	pretrained or not
progress	to see progress bar or not

**Details**

"Very Deep Convolutional Networks For Large-Scale Image Recognition" <<https://arxiv.org/pdf/1409.1556.pdf>>

**Value**

model

---

vision

*Vision module*


---

**Description**

Vision module

**Usage**

```
vision()
```

**Value**

None

---

vleaky_relu	<i>Vleaky_relu</i>
-------------	--------------------

---

**Description**

'F\$leaky\_relu' with 0.3 slope

**Usage**

```
vleaky_relu(input, inplace = TRUE)
```

**Arguments**

input	inputs
inplace	inplace or not

**Value**

None

---

Voice	<i>Voice</i>
-------	--------------

---

**Description**

Voice

**Usage**

```
Voice(
  sample_rate = 16000,
  n_fft = 1024,
  win_length = NULL,
  hop_length = 128,
  f_min = 50,
  f_max = 8000,
  pad = 0,
  n_mels = 128,
  window_fn = torch()$hann_window,
  power = 2,
  normalized = FALSE,
  kwargs = NULL,
  mel = TRUE,
  to_db = TRUE
)
```



**Arguments**

sample_rate	sample rate
n_fft	number of fast fourier transforms
win_length	windowing length
hop_length	hopping length
f_min	minimum frequency
f_max	maximum frequency
pad	padding mode
n_mels	number of mel-spectrograms
window_fn	window function
power	power
normalized	normalized or not
wkwargs	additional arguments
mel	mel-spectrogram or not
to_db	to decibels

**Value**

None

---

wandb

*Wandb module*

---

**Description**

Wandb module

**Usage**

wandb()

**Value**

None

---

WandbCallback

*WandbCallback*


---

**Description**

Saves model topology, losses & metrics

**Usage**

```
WandbCallback(
    log = "gradients",
    log_preds = TRUE,
    log_model = TRUE,
    log_dataset = FALSE,
    dataset_name = NULL,
    valid_dl = NULL,
    n_preds = 36,
    seed = 12345,
    reorder = TRUE
)
```

**Arguments**

log	"gradients" (default), "parameters", "all" or None. Losses & metrics are always logged.
log_preds	whether we want to log prediction samples (default to True).
log_model	whether we want to log our model (default to True). This also requires Save-ModelCallback.
log_dataset	Options: - False (default) - True will log folder referenced by learn.dls.path. - a path can be defined explicitly to reference which folder to log. Note: subfolder "models" is always ignored.
dataset_name	name of logged dataset (default to folder name).
valid_dl	DataLoaders containing items used for prediction samples (default to random items from learn.dls.valid).
n_preds	number of logged predictions (default to 36).
seed	used for defining random samples.
reorder	reorder or not

**Value**

None

---

Warp

*Warp*

---

### Description

Apply perspective warping with ‘magnitude’ and ‘p’ on a batch of matrices

### Usage

```
Warp(  
  magnitude = 0.2,  
  p = 0.5,  
  draw_x = NULL,  
  draw_y = NULL,  
  size = NULL,  
  mode = "bilinear",  
  pad_mode = "reflection",  
  batch = FALSE,  
  align_corners = TRUE  
)
```

### Arguments

magnitude	magnitude
p	probability
draw_x	draw x
draw_y	draw y
size	size
mode	mode
pad_mode	padding mode
batch	batch
align_corners	align corners

### Value

None

---

waterfall_plot	<i>Waterfall_plot</i>
----------------	-----------------------

---

**Description**

Plots an explanation of a single prediction as a waterfall plot. Accepts a `row_index` and `class_id`.

**Usage**

```
waterfall_plot(object, row_idx = NULL, class_id = 0, dpi = 200, ...)
```

**Arguments**

<code>object</code>	ShapInterpretation object
<code>row_idx</code>	is the index of the row chosen in <code>test_data</code> to be analyzed, which defaults to zero.
<code>class_id</code>	Accepts a <code>class_id</code> which is used to indicate the class of interest for a classification model. It can either be an int or str representation for a class of choice.
<code>dpi</code>	dots per inch
<code>...</code>	additional arguments

**Value**

None

---

WeightDropout	<i>WeightDropout</i>
---------------	----------------------

---

**Description**

A module that wraps another layer in which some weights will be replaced by 0 during training.

**Usage**

```
WeightDropout(module, weight_p, layer_names = "weight_hh_l0")
```

**Arguments**

<code>module</code>	module
<code>weight_p</code>	weight_p
<code>layer_names</code>	layer_names

**Value**

None

---

 WeightedDL

*WeightedDL*


---

**Description**

Transformed ‘DataLoader‘

**Usage**

```

WeightedDL(
  dataset = NULL,
  bs = NULL,
  wgts = NULL,
  shuffle = FALSE,
  num_workers = NULL,
  verbose = FALSE,
  do_setup = TRUE,
  pin_memory = FALSE,
  timeout = 0,
  batch_size = NULL,
  drop_last = FALSE,
  indexed = NULL,
  n = NULL,
  device = NULL,
  persistent_workers = FALSE
)

```

**Arguments**

dataset	dataset
bs	bs
wgts	weights
shuffle	shuffle
num_workers	number of workers
verbose	verbose
do_setup	do_setup
pin_memory	pin_memory
timeout	timeout
batch_size	batch_size
drop_last	drop_last
indexed	indexed
n	n
device	device
persistent_workers	persistent_workers

**Value**

None

---

weight_decay	<i>Weight_decay</i>
--------------	---------------------

---

**Description**

Weight decay as decaying 'p' with 'lr\*wd'

**Usage**

```
weight_decay(p, lr, wd, do_wd = TRUE, ...)
```

**Arguments**

p	p
lr	learning rate
wd	weight decay
do_wd	do_wd
...	additional args to pass

**Value**

None

**Examples**

```
## Not run:

tst_param = function(val, grad = NULL) {
  "Create a tensor with `val` and a gradient of `grad` for testing"
  res = tensor(val) %>% float()

  if(is.null(grad)) {
    grad = tensor(val / 10)
  } else {
    grad = tensor(grad)
  }

  res$grad = grad %>% float()
  res
}

p = tst_param(1., 0.1)
weight_decay(p, 1., 0.1)
```

## End(Not run)

---

win\_abdoment\_soft      *Abdomen soft*

---

**Description**

Abdomen soft

**Usage**

win\_abdoment\_soft()

**Value**

list

---

win\_brain              *Brain*

---

**Description**

Brain

**Usage**

win\_brain()

**Value**

list

---

win\_brain\_bone        *Brain bone*

---

**Description**

Brain bone

**Usage**

win\_brain\_bone()

**Value**

list

---

win_brain_soft	<i>Brain soft</i>
----------------	-------------------

---

**Description**

Brain soft

**Usage**

```
win_brain_soft()
```

**Value**

list

---

win_liver	<i>Liver</i>
-----------	--------------

---

**Description**

Liver

**Usage**

```
win_liver()
```

**Value**

list

---

win_lungs	<i>Lungs</i>
-----------	--------------

---

**Description**

Lungs

**Usage**

```
win_lungs()
```

**Value**

list



---

win_mediastinum	<i>Mediastinum</i>
-----------------	--------------------

---

**Description**

Mediastinum

**Usage**

win\_mediastinum()

**Value**

list

---

win_spine_bone	<i>Spine bone</i>
----------------	-------------------

---

**Description**

Spine bone

**Usage**

win\_spine\_bone()

**Value**

list

---

win_spine_soft	<i>Spine soft</i>
----------------	-------------------

---

**Description**

Spine soft

**Usage**

win\_spine\_soft()

**Value**

list

---

win_stroke	<i>Stroke</i>
------------	---------------

---

**Description**

Stroke

**Usage**

win\_stroke()

**Value**

list

---

win_subdural	<i>Subdural</i>
--------------	-----------------

---

**Description**

Subdural

**Usage**

win\_subdural()

**Value**

list

---

xla	<i>XLA</i>
-----	------------

---

**Description**

XLA

**Usage**

xla()

**Value**

None

---

XResNet	<i>XResNet</i>
---------	----------------

---

**Description**

A sequential container.

**Usage**

```
XResNet(block, expansion, layers, c_in = 3, c_out = 1000, ...)
```

**Arguments**

block	the blocks to pass to XResNet
expansion	argument for inputs and filters
layers	the layers to pass to XResNet
c_in	number of inputs
c_out	number of outputs
...	additional arguments

---

xresnet101	<i>Xresnet101</i>
------------	-------------------

---

**Description**

Load model architecture

**Usage**

```
xresnet101(...)
```

**Arguments**

...	parameters to pass
-----	--------------------

**Value**

model

---

`xresnet152`*Xresnet152*

---

**Description**

Load model architecture

**Usage**`xresnet152(...)`**Arguments**

... parameters to pass

**Value**model

---

`xresnet18`*Xresnet18*

---

**Description**

Load model architecture

**Usage**`xresnet18(...)`**Arguments**

... parameters to pass

**Value**

model

---

xresnet18_deep	<i>Xresnet18_deep</i>
----------------	-----------------------

---

**Description**

Load model architecture

**Usage**

xresnet18\_deep(...)

**Arguments**

... parameters to pass

**Value**

model

---

xresnet18_deeper	<i>Xresnet18_deeper</i>
------------------	-------------------------

---

**Description**

Load model architecture

**Usage**

xresnet18\_deeper(...)

**Arguments**

... parameters to pass

**Value**

model

---

`xresnet34`*Xresnet34*

---

**Description**

Load model architecture

**Usage**`xresnet34(...)`**Arguments**`...` parameters to pass**Value**model

---

`xresnet34_deep`*Xresnet34\_deep*

---

**Description**

Load model architecture

**Usage**`xresnet34_deep(...)`**Arguments**`...` parameters to pass**Value**

model

---

xresnet34\_deeper      *Xresnet34\_deeper*

---

**Description**

Load model architecture

**Usage**

xresnet34\_deeper(...)

**Arguments**

...      parameters to pass

**Value**

model

---

xresnet50      *Xresnet50*

---

**Description**

Load model architecture

**Usage**

xresnet50(...)

**Arguments**

...      parameters to pass

**Value**

model

---

xresnet50_deep	<i>Xresnet50_deep</i>
----------------	-----------------------

---

**Description**

Load model architecture

**Usage**

```
xresnet50_deep(...)
```

**Arguments**

... parameters to pass

**Value**

model

---

xresnet50_deeper	<i>Xresnet50_deeper</i>
------------------	-------------------------

---

**Description**

Load model architecture

**Usage**

```
xresnet50_deeper(...)
```

**Arguments**

... parameters to pass

**Value**

model



---

xresnext101

*xresnext101*

---

**Description**

Load model architecture

**Usage**

xresnext101(...)

**Arguments**

... parameters to pass

**Value**

model

---

xresnext18

*xresnext18*

---

**Description**

Load model architecture

**Usage**

xresnext18(...)

**Arguments**

... parameters to pass

**Value**

model

---

`xresnext34`*xresnext34*

---

**Description**

Load model architecture

**Usage**`xresnext34(...)`**Arguments**

... parameters to pass

**Value**model

---

`xresnext50`*xresnext50*

---

**Description**

Load model architecture

**Usage**`xresnext50(...)`**Arguments**

... parameters to pass

**Value**

model

---

`xsenet154`*xsenet154*

---

**Description**

Load model architecture

**Usage**`xsenet154(...)`**Arguments**

... parameters to pass

**Value**model

---

`xse_resnet101`*xse\_resnet101*

---

**Description**

Load model architecture

**Usage**`xse_resnet101(...)`**Arguments**

... parameters to pass

**Value**

model

---

`xse_resnet152`*xse\_resnet152*

---

**Description**

Load model architecture

**Usage**`xse_resnet152(...)`**Arguments**`...` parameters to pass**Value**model

---

`xse_resnet18`*xse\_resnet18*

---

**Description**

Load model architecture

**Usage**`xse_resnet18(...)`**Arguments**`...` parameters to pass**Value**

model

---

`xse_resnet34``xse_resnet34`

---

**Description**

Load model architecture

**Usage**

```
xse_resnet34(...)
```

**Arguments**

```
...           parameters to pass
```

**Value**

model

---

`xse_resnet50``xse_resnet50`

---

**Description**

Load model architecture

**Usage**

```
xse_resnet50(...)
```

**Arguments**

```
...           parameters to pass
```

**Value**

model

---

xse_resnext101	<i>xse_resnext101</i>
----------------	-----------------------

---

**Description**

Load model architecture

**Usage**

```
xse_resnext101(...)
```

**Arguments**

... parameters to pass

**Value**

model

---

xse_resnext18	<i>xse_resnext18</i>
---------------	----------------------

---

**Description**

Load model architecture

**Usage**

```
xse_resnext18(...)
```

**Arguments**

... parameters to pass

**Value**

model

---

*xse\_resnext18\_deep*     *xse\_resnext18\_deep*

---

**Description**

Load model architecture

**Usage**

`xse_resnext18_deep(...)`

**Arguments**

...            parameters to pass

**Value**

model

---

*xse\_resnext18\_deeper*     *xse\_resnext18\_deeper*

---

**Description**

Load model architecture

**Usage**

`xse_resnext18_deeper(...)`

**Arguments**

...            parameters to pass

**Value**

model

---

`xse_resnext34``xse_resnext34`

---

**Description**

Load model architecture

**Usage**

```
xse_resnext34(...)
```

**Arguments**

... parameters to pass

**Value**

model

---

`xse_resnext34_deep``xse_resnext34_deep`

---

**Description**

Load model architecture

**Usage**

```
xse_resnext34_deep(...)
```

**Arguments**

... parameters to pass

**Value**

model



---

`xse_resnext34_deeper`    *xse\_resnext34\_deeper*

---

**Description**

Load model architecture

**Usage**

`xse_resnext34_deeper(...)`

**Arguments**

...                    parameters to pass

**Value**

model

---

`xse_resnext50`                    *xse\_resnext50*

---

**Description**

Load model architecture

**Usage**

`xse_resnext50(...)`

**Arguments**

...                    parameters to pass

**Value**

model

---

xse\_resnext50\_deep     *xse\_resnext50\_deep*

---

**Description**

Load model architecture

**Usage**

xse\_resnext50\_deep(...)

**Arguments**

...                    parameters to pass

**Value**

model

---

xse\_resnext50\_deeper     *xse\_resnext50\_deeper*

---

**Description**

Load model architecture

**Usage**

xse\_resnext50\_deeper(...)

**Arguments**

...                    parameters to pass

**Value**

model

---

 zoom

*Zoom*


---

**Description**

Zoom

**Usage**

```
zoom(img, ratio)
```

**Arguments**

img	image files
ratio	ratio

**Value**

image

---

 Zoom\_

*Zoom*


---

**Description**

Apply a random zoom of at most ‘max\_zoom’ with probability ‘p’ to a batch of images

**Usage**

```
Zoom_(
  min_zoom = 1,
  max_zoom = 1.1,
  p = 0.5,
  draw = NULL,
  draw_x = NULL,
  draw_y = NULL,
  size = NULL,
  mode = "bilinear",
  pad_mode = "reflection",
  batch = FALSE,
  align_corners = TRUE
)
```

**Arguments**

min_zoom	minimum zoom
max_zoom	maximum zoom
p	probability
draw	draw
draw_x	draw x
draw_y	draw y
size	size
mode	mode
pad_mode	pad mode
batch	batch
align_corners	align corners or not

**Value**

None

zoom\_mat

*Zoom\_mat***Description**

Return a random zoom matrix with ‘max\_zoom’ and ‘p’

**Usage**

```
zoom_mat(
  x,
  min_zoom = 1,
  max_zoom = 1.1,
  p = 0.5,
  draw = NULL,
  draw_x = NULL,
  draw_y = NULL,
  batch = FALSE
)
```

**Arguments**

x	tensor
min_zoom	minimum zoom
max_zoom	maximum zoom
p	probability

draw	draw
draw_x	draw x
draw_y	draw y
batch	batch

**Value**

None

---

&.fastai.torch\_core.TensorMask  
*Logical\_and*

---

**Description**

Logical\_and

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'
x & y
```

**Arguments**

x	tensor
y	tensor

**Value**

tensor

---

%% *Fastai assignment*

---

**Description**

The assignment has to be used for safe modification of the values inside tensors/layers

**Usage**

```
left %% right
```

**Arguments**

left	left side object
right	right side object

**Value**

None

---

%%.fastai.torch\_core.TensorMask  
*Floor mod*

---

**Description**

Floor mod

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
x %% y
```

**Arguments**

x	tensor
y	tensor

**Value**

tensor

---

%%.fastai.torch\_core.TensorMask  
*Floor divide*

---

**Description**

Floor divide

**Usage**

```
## S3 method for class 'fastai.torch_core.TensorMask'  
x %%/ y
```

**Arguments**

x	tensor
y	tensor

**Value**

tensor

---

`^.fastai.torch_core.TensorMask`  
*Pow*

---

### Description

Pow

### Usage

```
## S3 method for class 'fastai.torch_core.TensorMask'  
a ^ b
```

### Arguments

a	tensor
b	tensor

### Value

tensor

# Index

!.fastai.torch\_core.TensorMask  
    (not\_\_mask), 375  
!.torch.Tensor (logical\_not\_), 340  
!=.fastai.torch\_core.TensorMask  
    (not\_equal\_to\_mask\_), 375  
!=.torch.Tensor (not\_equal\_to), 374  
\*.fastai.torch\_core.TensorMask, 22  
\*.torch.Tensor (multiplygit add -A &&  
    git commit -m 'staging all  
    files'), 367  
+.fastai.torch\_core.TensorMask, 22  
+.torch.Tensor (add), 35  
+.torch.nn.modules.container.Sequential,  
    23  
-.fastai.torch\_core.TensorMask  
    (sub\_mask), 488  
-.torch.Tensor (sub), 487  
/.fastai.torch\_core.TensorMask, 23  
/.torch.Tensor (div), 131  
<.fastai.torch\_core.TensorMask, 24  
<.torch.Tensor (less), 327  
<=.fastai.torch\_core.TensorMask, 24  
<=.torch.Tensor (less\_or\_equal), 327  
==.fastai.torch\_core.TensorImage, 25  
==.fastai.torch\_core.TensorMask, 25  
==.torch.Tensor, 26  
>.fastai.torch\_core.TensorMask, 26  
>.torch.Tensor (greater), 196  
>=.fastai.torch\_core.TensorMask, 27  
>=.torch.Tensor (greater\_or\_equal), 197  
%/.torch.Tensor (floor\_div), 168  
%/.torch.Tensor (floor\_mod), 168  
&.fastai.torch\_core.TensorMask, 605  
&.torch.Tensor (logical\_and), 339  
%/.fastai.torch\_core.TensorMask, 606  
%/.fastai.torch\_core.TensorMask, 606  
%f%, 605  
^.fastai.torch\_core.TensorMask, 607  
^.torch.Tensor (pow), 403  
abs, 27  
abs.fastai.torch\_core.TensorMask, 28  
AccumMetric, 28  
accuracy, 29  
accuracy\_multi, 30  
accuracy\_thresh\_expand, 30  
Adam, 31  
adam\_step, 31  
adaptive\_pool, 34  
AdaptiveAvgPool, 32  
AdaptiveConcatPool1d, 32  
AdaptiveConcatPool2d, 33  
AdaptiveGANSwitcher, 33  
AdaptiveLoss, 34  
add, 35  
add\_cyclic\_datepart, 36  
add\_datepart, 37  
AddChannels, 35  
AddNoise, 36  
affine\_coord, 38  
affine\_mat, 39  
AffineCoordTfm, 37  
alexnet, 39  
apply\_perspective, 40  
APScoreBinary, 40  
APScoreMulti, 41  
as\_array, 42  
aspect, 42  
audio\_extensions, 48  
AudioBlock, 43  
AudioBlock\_from\_folder, 43  
AudioGetter, 44  
AudioPadType, 45  
AudioSpectrogram, 45  
AudioTensor, 45  
AudioTensor\_create, 46  
AudioToMFCC, 47  
AudioToMFCC\_from\_cfg, 47  
AudioToSpec\_from\_cfg, 48



- aug\_transforms, 49
- AutoConfig, 50
- average\_grad, 51
- average\_sqr\_grad, 51
- AvgLoss, 52
- AvgPool, 52
- AvgSmoothLoss, 53
- AWD\_LSTM, 53
- awd\_lstm\_clas\_split, 54
- awd\_lstm\_lm\_split, 55
- AWD\_QRNN, 55
  
- BalancedAccuracy, 56
- BaseLoss, 57
- BaseTokenizer, 57
- basic\_critic, 61
- basic\_generator, 61
- BasicMelSpectrogram, 58
- BasicMFCC, 59
- BasicSpectrogram, 60
- BatchNorm, 62
- BatchNorm1dFlat, 63
- bb\_pad, 65
- BBoxBlock, 63
- BBoxLabeler, 64
- BBoxLblBlock, 64
- BCELossFlat, 66
- BCEWithLogitsLossFlat, 66
- blurr, 67
- BrierScore, 67
- BrierScoreMulti, 68
- bs\_find, 68
- bs\_finder, 69
- bt, 69
  
- calculate\_rouge, 70
- Callback, 70
- Cat, 71
- catalyst, 71
- catalyst\_model, 72
- Categorify, 72
- CategoryBlock, 73
- ceiling.fastai.torch\_core.TensorMask, 73
- ceiling.torch.Tensor(ceiling\_), 74
- ceiling\_, 74
- ChangeVolume, 74
- children\_and\_parameters, 75
- ClassificationInterpretation\_from\_learner, 75
- clean\_raw\_keys, 76
- clip\_remove\_empty, 76
- cm, 77
- cnn\_config, 77
- cnn\_learner, 78
- COCOMetric, 80
- COCOMetricType, 81
- CohenKappa, 81
- collab, 82
- collab\_learner, 84
- CollabDataLoaders\_from\_dblock, 82
- CollabDataLoaders\_from\_df, 83
- CollectDataCallback, 86
- colors, 86
- ColReader, 87
- ColSplitter, 87
- combined\_flat\_anneal, 88
- competition\_download\_file, 89
- competition\_download\_files, 90
- competition\_leaderboard\_download, 91
- competition\_list\_files, 91
- competition\_submit, 92
- competitions\_list, 88
- Contrast, 92
- conv\_norm\_lr, 95
- ConvLayer, 93
- convT\_norm\_relu, 94
- CorpusBLEUMetric, 96
- cos.fastai.torch\_core.TensorMask, 96
- cos.torch.Tensor(cos\_), 98
- cos\_, 98
- cosh.fastai.torch\_core.TensorMask, 97
- cosh.torch.Tensor(cosh\_), 97
- cosh\_, 97
- crap, 98
- crappifier, 99
- create\_body, 99
- create\_cnn\_model, 100
- create\_fcn, 101
- create\_head, 102
- create\_inception, 103
- create\_mlp, 103
- create\_resnet, 104
- create\_unet\_model, 105
- CropPad, 106
- CropTime, 106

- CrossEntropyLossFlat, 107
- CSVLogger, 107
- CudaCallback, 108
- custom\_loss, 109
- CutMix, 109
- cutout\_gaussian, 110
- cycle\_learner, 112
- CycleGAN, 110
- CycleGANLoss, 111
- CycleGANTrainer, 112
  
- Data\_Loaders, 116
- DataBlock, 113
- dataloaders, 114
- Datasets, 115
- dcmread, 117
- debias, 117
- Debugger, 118
- decision\_plot, 118
- decode\_spec\_tokens, 119
- default\_split, 119
- Delta, 120
- denormalize\_imagenet, 120
- densenet121, 121
- densenet161, 121
- densenet169, 122
- densenet201, 122
- DenseResBlock, 123
- dependence\_plot, 124
- DeterministicDihedral, 125
- DeterministicDraw, 125
- DeterministicFlip, 126
- detuplify\_pg, 126
- Dice, 127
- Dicom, 127
- dicom\_windows, 127
- Dihedral, 128
- dihedral\_mat, 129
- DihedralItem, 129
- dim, 130
- dim.fastai.torch\_core.TensorMask, 130
- discriminator, 131
- div, 131
- DownmixMono, 132
- dropout\_mask, 132
- dummy\_eval, 133
- DynamicUnet, 133
  
- EarlyStoppingCallback, 134
  
- efficientdet\_infer\_dl, 135
- efficientdet\_learner, 135
- efficientdet\_model, 136
- efficientdet\_predict\_dl, 136
- efficientdet\_train\_dl, 137
- efficientdet\_valid\_dl, 137
- emb\_sz\_rule, 139
- Embedding, 138
- EmbeddingDropout, 138
- error\_rate, 139
- exp, 140
- exp.fastai.torch\_core.TensorMask, 140
- exp\_rmsep, 143
- ExplainedVariance, 141
- expm1, 141
- expm1.fastai.torch\_core.TensorMask, 142
- export\_generator, 142
  
- F1Score, 143
- F1ScoreMulti, 144
- fa\_collate, 150
- fa\_convert, 150
- fastai\_version, 145
- fastaudio, 145
- faster\_rcnn\_infer\_dl, 145
- faster\_rcnn\_learner, 146
- faster\_rcnn\_model, 147
- faster\_rcnn\_predict\_dl, 147
- faster\_rcnn\_train\_dl, 148
- faster\_rcnn\_valid\_dl, 149
- fastinf, 149
- FBeta, 151
- FBetaMulti, 151
- FetchPredsCallback, 152
- FileSplitter, 153
- FillMissing, 153
- FillStrategy\_COMMON, 154
- FillStrategy\_CONSTANT, 155
- FillStrategy\_MEDIAN, 155
- find\_coeffs, 155
- fine\_tune, 156
- fit.fastai.learner.Learner, 157
- fit.fastai.tabular.learner.TabularLearner, 157
- fit.fastai.vision.gan.GANLearner, 158
- fit\_flat\_cos, 158
- fit\_flat\_lin, 159
- fit\_one\_cycle, 160

- fit\_sgdr, 161
- fix\_fit, 162
- fix\_html, 163
- FixedGANSwitcher, 162
- Flatten, 163
- flatten\_check, 164
- flatten\_model, 164
- Flip, 165
- flip\_mat, 166
- FlipItem, 165
- float, 166
- floor.fastai.torch\_core.TensorMask, 167
- floor.torch.Tensor (floor\_), 167
- floor\_, 167
- floor\_div, 168
- floor\_mod, 168
- fmodule, 169
- FolderDataset, 169
- force\_plot, 170
- foreground\_acc, 170
- forget\_mult\_CPU, 171
- ForgetMultGPU, 171
- freeze, 172
- FuncSplitter, 172
- fView, 173
  
- gan\_critic, 178
- gan\_loss\_from\_func, 179
- GANDiscriminativeLR, 173
- GANLearner\_from\_learners, 174
- GANLearner\_wgan, 175
- GANLoss, 177
- GANModule, 177
- GANTrainer, 178
- GatherPredsCallback, 179
- gauss\_blur2d, 180
- generate\_noise, 180
- get\_annotations, 181
- get\_audio\_files, 182
- get\_bias, 182
- get\_c, 183
- get\_confusion\_matrix, 184
- get\_data\_loaders, 184
- get\_dcm\_matrix, 185
- get\_dicom\_files, 186
- get\_dls, 186
- get\_emb\_sz, 187
- get\_files, 188
- get\_grid, 189
- get\_hf\_objects, 190
- get\_image\_files, 190
- get\_language\_model, 191
- get\_preds\_cyclegan, 192
- get\_text\_classifier, 192
- get\_text\_files, 193
- get\_weights, 194
- GradientAccumulation, 195
- GrandparentSplitter, 195
- grayscale, 196
- greater, 196
- greater\_or\_equal, 197
  
- HammingLoss, 197
- HammingLossMulti, 198
- has\_params, 198
- has\_pool\_type, 199
- helper, 199
- HF\_ARCHITECTURES, 199
- HF\_BaseInput, 200
- HF\_BaseModelCallback, 200
- HF\_BaseModelWrapper, 201
- HF\_BeforeBatchTransform, 201
- HF\_CausalLMBeforeBatchTransform, 202
- HF\_load\_dataset, 203
- HF\_QABatchTransform, 205
- HF\_QABeforeBatchTransform, 206
- HF\_QstAndAnsModelCallback, 207
- HF\_QuestionAnswerInput, 207
- hf\_splitter, 208
- HF\_SummarizationBeforeBatchTransform, 208
- HF\_SummarizationInput, 209
- HF\_SummarizationModelCallback, 210
- HF\_TASKS\_ALL, 210
- HF\_TASKS\_AUTO, 211
- HF\_Text2TextAfterBatchTransform, 211
- HF\_Text2TextBlock, 212
- HF\_TextBlock, 212
- HF\_TokenCategorize, 213
- HF\_TokenCategoryBlock, 213
- HF\_TokenClassBeforeBatchTransform, 214
- HF\_TokenClassInput, 215
- HF\_TokenTensorCategory, 215
- Hook, 215
- hook\_output, 218
- hook\_outputs, 218
- HookCallback, 216

- Hooks, 217
- hsv2rgb, 219
- Hue, 219
- hug, 220
  
- icevision, 220
- icevision\_Adapter, 220
- icevision\_aug\_tfms, 221
- icevision\_BasicIAATransform, 222
- icevision\_BasicTransform, 222
- icevision\_Blur, 223
- icevision\_ChannelDropout, 223
- icevision\_ChannelShuffle, 224
- icevision\_CLAHE, 225
- icevision\_ClassMap, 225
- icevision\_CoarseDropout, 226
- icevision\_ColorJitter, 227
- icevision\_Compose, 228
- icevision\_Crop, 229
- icevision\_CropNonEmptyMaskIfExists, 229
  
- icevision\_Cutout, 230
- icevision\_Dataset, 231
- icevision\_Dataset\_from\_images, 232
- icevision\_Downsacle, 233
- icevision\_DualIAATransform, 234
- icevision\_DualTransform, 234
- icevision\_ElasticTransform, 235
- icevision\_Equalize, 236
- icevision\_FancyPCA, 237
- icevision\_FDA, 238
- icevision\_FixedSplitter, 239
- icevision\_Flip, 239
- icevision\_FromFloat, 240
- icevision\_GaussianBlur, 241
- icevision\_GaussNoise, 242
- icevision\_GlassBlur, 242
- icevision\_GridDistortion, 243
- icevision\_GridDropout, 245
- icevision\_HistogramMatching, 246
- icevision\_HorizontalFlip, 247
- icevision\_HueSaturationValue, 248
- icevision\_IAAAdditiveGaussianNoise, 249
  
- icevision\_IAAAffine, 249
- icevision\_IACropAndPad, 251
- icevision\_IAAEmboss, 251
- icevision\_IAAFliplr, 252
- icevision\_IAAFlipud, 253
  
- icevision\_IAAPerspective, 253
- icevision\_IAAPiecewiseAffine, 254
- icevision\_IAASharpen, 255
- icevision\_IAASuperpixels, 256
- icevision\_ImageCompression, 257
- icevision\_ImageOnlyIAATransform, 258
- icevision\_ImageOnlyTransform, 258
- icevision\_InvertImg, 259
- icevision\_ISONoise, 259
- icevision\_JpegCompression, 260
- icevision\_LongestMaxSize, 261
- icevision\_MaskDropout, 262
- icevision\_MedianBlur, 263
- icevision\_MotionBlur, 263
- icevision\_MultiplicativeNoise, 264
- icevision\_Normalize, 265
- icevision\_OpticalDistortion, 266
- icevision\_PadIfNeeded, 267
- icevision\_parse, 268
- icevision\_Posterize, 269
- icevision\_RandomBrightnessContrast, 269
  
- icevision\_RandomContrast, 270
- icevision\_RandomCrop, 271
- icevision\_RandomCropNearBBox, 272
- icevision\_RandomFog, 272
- icevision\_RandomGamma, 273
- icevision\_RandomGridShuffle, 274
- icevision\_RandomRain, 275
- icevision\_RandomResizedCrop, 276
- icevision\_RandomRotate90, 277
- icevision\_RandomScale, 277
- icevision\_RandomShadow, 278
- icevision\_RandomSizedBBoxSafeCrop, 279
- icevision\_RandomSizedCrop, 280
- icevision\_RandomSnow, 281
- icevision\_RandomSplitter, 282
- icevision\_RandomSunFlare, 283
- icevision\_read\_bgr\_image, 284
- icevision\_read\_rgb\_image, 284
- icevision\_Resize, 285
- icevision\_resize\_and\_pad, 285
- icevision\_RGBShift, 286
- icevision\_Rotate, 287
- icevision\_ShiftScaleRotate, 288
- icevision\_SingleSplitSplitter, 289
- icevision\_SmallestMaxSize, 290
- icevision\_Solarize, 291

- icevision\_ToFloat, 291
- icevision\_ToGray, 292
- icevision\_ToSepia, 293
- icevision\_Transpose, 293
- icevision\_VerticalFlip, 294
- icnr\_init, 295
- IDMap, 295
- Image, 296
- image2tensor, 296
- Image\_create, 307
- Image\_open, 308
- Image\_resize, 308
- ImageBlock, 297
- ImageBW\_create, 297
- ImageDataLoaders\_from\_csv, 298
- ImageDataLoaders\_from\_dblock, 299
- ImageDataLoaders\_from\_df, 300
- ImageDataLoaders\_from\_folder, 301
- ImageDataLoaders\_from\_lists, 302
- ImageDataLoaders\_from\_name\_re, 303
- ImageDataLoaders\_from\_path\_func, 305
- ImageDataLoaders\_from\_path\_re, 306
- imagenet\_stats, 307
- in\_channels, 314
- InceptionModule, 309
- IndexSplitter, 309
- init, 310
- init\_default, 310
- init\_linear, 311
- install\_fastai, 311
- InstanceNorm, 312
- IntToFloatTensor, 313
- InvisibleTensor, 313
- is\_rmarkdown, 314
  
- Jaccard, 315
- JaccardCoeff, 315
- JaccardMulti, 316
  
- kg, 316
  
- L, 317
- L1LossFlat, 317
- l2\_reg, 318
- LabeledBBox, 319
- LabelSmoothingCrossEntropy, 319
- LabelSmoothingCrossEntropyFlat, 320
- Lamb, 320
- lamb\_step, 321
- Lambda, 321
- language\_model\_learner, 322
- Larc, 323
- larc\_layer\_lr, 324
- larc\_step, 324
- layer\_info, 325
- Learner, 325
- length, 326
- length.fastai.torch\_core.TensorMask, 326
- less, 327
- less\_or\_equal, 327
- LightingTfm, 328
- LinBnDrop, 328
- LinearDecoder, 329
- LitModel, 329
- LMDataLoader, 330
- LMLearner, 331
- LMLearner\_predict, 332
- load\_dataset, 334
- load\_ignore\_keys, 334
- load\_image, 335
- load\_learner, 335
- load\_model\_text, 336
- load\_pre\_models, 336
- load\_tokenized\_csv, 337
- loaders, 333
- log, 337
- log.fastai.torch\_core.TensorMask, 338
- log1p, 338
- log1p.fastai.torch\_core.TensorMask, 339
- logical\_and, 339
- logical\_not\_, 340
- logical\_or, 340
- login, 341
- Lookahead, 341
- LossMetric, 342
- lr\_find, 342
  
- mae, 343
- make\_vocab, 343
- mask2bbox, 344
- Mask\_create, 346
- mask\_from\_blur, 347
- mask\_rcnn\_infer\_dl, 347
- mask\_rcnn\_learner, 348
- mask\_rcnn\_model, 348
- mask\_rcnn\_predict\_dl, 349

mask\_rcnn\_train\_dl, 350  
mask\_rcnn\_valid\_dl, 350  
mask\_tensor, 351  
MaskBlock, 344  
masked\_concat\_pool, 345  
MaskFreq, 345  
MaskTime, 346  
match\_embeds, 351  
MatthewsCorrCoef, 352  
MatthewsCorrCoefMulti, 352  
max, 353  
max.fastai.torch\_core.TensorMask, 353  
MaxPool, 354  
maybe\_unsqueeze, 354  
MCDropoutCallback, 355  
mean.fastai.torch\_core.TensorMask, 355  
mean.torch.Tensor, 356  
medical, 356  
MergeLayer, 357  
metrics, 357  
migrating\_ignite, 357  
migrating\_lightning, 358  
migrating\_pytorch, 358  
min, 358  
min.fastai.torch\_core.TensorMask, 359  
mish, 359  
Mish\_, 360  
MishJitAutoFn, 360  
MixHandler, 361  
MixUp, 361  
model\_sizes, 362  
ModelResetter, 362  
Module, 363  
Module\_test, 363  
momentum\_step, 363  
most\_confused, 364  
mse, 364  
MSELossFlat, 365  
msle, 366  
MultiCategorize, 366  
MultiCategoryBlock, 367  
multiplygit add -A && git commit -m  
    'staging all files', 367  
MultiTargetLoss, 368  
  
n\_px, 377  
narrow, 368  
Net, 369  
nn, 369  
nn\_loss, 370  
nn\_module, 370  
NoiseColor, 371  
NoneReduce, 371  
noop, 372  
norm\_apply\_denorm, 374  
Normalize, 372  
Normalize\_from\_stats, 373  
NormalizeTS, 373  
not\_\_mask, 375  
not\_equal\_to, 374  
not\_equal\_to\_mask\_, 375  
num\_features\_model, 376  
Numericalize, 376  
  
OldRandomCrop, 377  
one\_batch, 378  
OpenAudio, 378  
optim\_metric, 380  
Optimizer, 379  
OptimWrapper, 379  
or\_mask, 380  
os, 381  
os\_envIRON\_tpu, 381  
  
pad\_conv\_norm\_relu, 382  
pad\_input, 383  
pad\_input\_chunk, 383  
parallel, 384  
parallel\_tokenize, 384  
params, 385  
ParamScheduler, 385  
parent\_label, 386  
parsers\_AreasMixin, 386  
parsers\_BBoxesMixin, 387  
parsers\_FasterRCNN, 387  
parsers\_FilepathMixin, 388  
parsers\_ImageidMixin, 388  
parsers\_IsCrowdsMixin, 389  
parsers\_LabelsMixin, 389  
parsers\_MaskRCNN, 390  
parsers\_MasksMixin, 390  
parsers\_SizeMixin, 391  
parsers\_voc, 391  
partial, 392  
PartialDL, 392  
PartialLambda, 393  
pca, 394  
PearsonCorrCoef, 394

- Perplexity, 395
- Pipeline, 396
- PixelShuffle\_ICNR, 396
- plot, 397
- plot\_bs\_find, 397
- plot\_confusion\_matrix, 398
- plot\_loss, 399
- plot\_lr\_find, 399
- plot\_top\_losses, 400
- PointBlock, 401
- PointScaler, 401
- PooledSelfAttention2d, 402
- PoolFlatten, 402
- PoolingLinearClassifier, 403
- pow, 403
- pre\_process\_squad, 408
- Precision, 404
- PrecisionMulti, 404
- predict.fastai.learner.Learner, 405
- predict.fastai.tabular.learner.TabularLearner, 406
- preplexity, 406
- preprocess\_audio\_folder, 407
- PreprocessAudio, 407
- print.fastai.learner.Learner, 409
- print.fastai.tabular.learner.TabularLearner, 409
- print.pydicom.dataset.FileDataset, 410
- py\_apply, 411
- python\_path, 410
  
- QHAdam, 411
- qhadam\_step, 412
- QRNN, 412
- QRNNLayer, 413
  
- R2Score, 414
- RAdam, 415
- radam\_step, 415
- RandomCrop, 416
- RandomErasing, 416
- RandomResizedCrop, 417
- RandomResizedCropGPU, 417
- RandomSplitter, 418
- RandPair, 418
- RandTransform, 419
- ranger, 419
- RatioResize, 420
- ReadTSBatch, 421
  
- Recall, 421
- RecallMulti, 422
- ReduceLRonPlateau, 422
- RegressionBlock, 423
- RemoveSilence, 424
- RemoveType, 424
- replace\_all\_caps, 425
- replace\_maj, 425
- replace\_rep, 426
- replace\_wrep, 426
- res\_block\_1d, 436
- Resample, 427
- ResBlock, 427
- reshape, 429
- Resize, 429
- resize\_max, 431
- ResizeBatch, 430
- ResizeSignal, 430
- ResNet, 431
- resnet101, 432
- resnet152, 433
- resnet18, 433
- resnet34, 434
- resnet50, 434
- resnet\_generator, 435
- ResnetBlock, 435
- RetinaNet, 437
- retinanet\_, 438
- RetinaNetFocalLoss, 437
- reverse\_text, 438
- rgb2hsv, 439
- rm\_useless\_spaces, 441
- rms\_prop\_step, 440
- rmse, 439
- RMSProp, 440
- RNNDropout, 442
- RNNRegularizer, 442
- RocAuc, 443
- RocAucBinary, 443
- RocAucMulti, 444
- Rotate, 445
- rotate\_mat, 446
- round, 446
- round.fastai.torch\_core.TensorMask, 447
  
- Saturation, 447
- SaveModelCallback, 448
- SchedCos, 448

- SchedExp, 449
- SchedLin, 449
- SchedNo, 450
- SchedPoly, 450
- SEBlock, 451
- SegmentationDataLoaders\_from\_label\_func, 451
- SelfAttention, 452
- SEModule, 453
- SentenceEncoder, 453
- SentencePieceTokenizer, 454
- SeparableBlock, 455
- sequential, 455
- SequentialEx, 456
- SequentialRNN, 456
- SEResNeXtBlock, 457
- set\_freeze\_model, 458
- set\_item\_pg, 458
- setup\_aug\_tfms, 457
- SGD, 459
- sgd\_step, 459
- SGRoll, 460
- shap, 461
- shape, 461
- ShapInterpretation, 462
- Shortcut, 463
- ShortEpochCallback, 463
- show, 464
- show\_array, 465
- show\_batch, 466
- show\_image, 467
- show\_images, 468
- show\_preds, 469
- show\_results, 470
- show\_samples, 470
- ShowCycleGANImgsCallback, 464
- ShowGraphCallback, 465
- sigmoid, 471
- sigmoid\_, 472
- sigmoid\_range, 473
- SigmoidRange, 472
- SignalCutout, 473
- SignalLoss, 474
- SignalShifter, 474
- SimpleCNN, 475
- SimpleSelfAttention, 475
- sin.fastai.torch\_core.TensorMask, 476
- sin.torch.Tensor (sin\_), 477
- sin\_, 477
- sinh.fastai.torch\_core.TensorMask, 476
- sinh.torch.Tensor (add), 35
- skm\_to\_fastai, 477
- slice, 478
- sort, 478
- sort.fastai.torch\_core.TensorMask, 479
- SortedDL, 479
- SpacyTokenizer, 480
- SpearmanCorrCoef, 481
- spec\_add\_spaces, 482
- SpectrogramTransformer, 482
- sqr, 483
- sqr.fastai.torch\_core.TensorMask, 483
- sqr.torch.Tensor (sqr), 483
- SqueezeNet, 484
- squeezenet1\_0, 484
- squeezenet1\_1, 485
- stack\_train\_valid, 486
- step\_stat, 486
- sub, 487
- sub\_mask, 488
- subplots, 487
- summarization\_splitter, 488
- summary.fastai.learner.Learner, 489
- summary.fastai.tabular.learner.TabularLearner, 489
- summary\_plot, 490
- swish, 490
- Swish\_, 491
- tabular, 491
- tabular\_config, 496
- tabular\_learner, 497
- TabularDataTable, 492
- TabularModel, 493
- TabularTS, 494
- TabularTSDataloader, 495
- tar\_extract\_at\_filename, 498
- tensor, 499
- TensorBBox, 499
- TensorBBox\_create, 500
- TensorImage, 500
- TensorImageBW, 501
- TensorMultiCategory, 501
- TensorPoint, 502
- TensorPoint\_create, 502
- TerminateOnNaNCallback, 503
- test\_loader, 503



text, 504  
text\_classifier\_learner, 514  
TextBlock, 504  
TextBlock\_from\_df, 505  
TextBlock\_from\_folder, 506  
TextDataLoaders\_from\_csv, 507  
TextDataLoaders\_from\_df, 508  
TextDataLoaders\_from\_folder, 510  
TextLearner, 511  
TextLearner\_load\_encoder, 512  
TextLearner\_load\_pretrained, 513  
TextLearner\_save\_encoder, 513  
Tfmdl, 515  
TfmdlLists, 516  
TfmResize, 517  
timm, 517  
timm\_learner, 518  
timm\_list\_models, 518  
tms, 519  
to\_bytes\_format, 528  
to\_image, 529  
to\_matrix, 529  
to\_thumb, 530  
to\_xla, 530  
tokenize1, 519  
tokenize\_csv, 522  
tokenize\_df, 523  
tokenize\_files, 523  
tokenize\_folder, 524  
tokenize\_texts, 525  
Tokenizer, 520  
Tokenizer\_from\_df, 520  
TokenizeWithRules, 521  
top\_k\_accuracy, 526  
torch, 527  
total\_params, 527  
ToTensor, 528  
TrackerCallback, 531  
train\_loader, 532  
trainable\_params, 531  
TrainEvalCallback, 532  
Transform, 533  
TransformBlock, 533  
transformers, 534  
TransformersDropOutput, 534  
TransformersTokenizer, 535  
trunc\_normal\_, 535  
TSBlock, 536  
TSDataLoaders\_from\_dfs, 536  
TSDataTable, 537  
TSeries, 538  
TSeries\_create, 539  
  
unet\_config, 541  
unet\_learner, 542  
UnetBlock, 539  
unfreeze, 542  
uniform\_blur2d, 543  
upit, 543  
URLs\_ADULT\_SAMPLE, 544  
URLs\_AG\_NEWS, 544  
URLs\_AMAZON\_REVIEWS\_POLARITY, 546  
URLs\_AMAZON\_REVIEWSAMAZON\_REVIEWS, 545  
URLs\_BIWI\_HEAD\_POSE, 546  
URLs\_CALTECH\_101, 547  
URLs\_CAMVID, 547  
URLs\_CAMVID\_TINY, 548  
URLs\_CARS, 548  
URLs\_CIFAR, 549  
URLs\_CIFAR\_100, 549  
URLs\_COCO\_TINY, 550  
URLs\_CUB\_200\_2011, 550  
URLs\_DBPEDIA, 551  
URLs\_DOGS, 551  
URLs\_FLOWERS, 552  
URLs\_FOOD, 552  
URLs\_HORSE\_2\_ZEBRA, 553  
URLs\_HUMAN\_NUMBERS, 553  
URLs\_IMAGENETTE, 554  
URLs\_IMAGENETTE\_160, 554  
URLs\_IMAGENETTE\_320, 555  
URLs\_IMAGEWOOF, 555  
URLs\_IMAGEWOOF\_160, 556  
URLs\_IMAGEWOOF\_320, 556  
URLs\_IMDB, 557  
URLs\_IMDB\_SAMPLE, 557  
URLs\_LSUN\_BEDROOMS, 558  
URLs\_ML\_SAMPLE, 558  
URLs\_MNIST, 559  
URLs\_MNIST\_SAMPLE, 559  
URLs\_MNIST\_TINY, 560  
URLs\_MNIST\_VAR\_SIZE\_TINY, 560  
URLs\_MOVIE\_LENS\_ML\_100k, 561  
URLs\_MT\_ENG\_FRA, 561  
URLs\_OPENAI\_TRANSFORMER, 562  
URLs\_PASCAL\_2007, 562  
URLs\_PASCAL\_2012, 563

URLs\_PETS, [563](#)  
URLs\_PLANET\_SAMPLE, [564](#)  
URLs\_PLANET\_TINY, [564](#)  
URLs\_S3\_COCO, [565](#)  
URLs\_S3\_IMAGE, [565](#)  
URLs\_S3\_IMAGELOC, [566](#)  
URLs\_S3\_MODEL, [566](#)  
URLs\_S3\_NLP, [567](#)  
URLs\_SIIM\_SMALL, [567](#)  
URLs\_SKIN\_LESION, [568](#)  
URLs\_SOGOOU\_NEWS, [568](#)  
URLs\_SPEAKERS10, [569](#)  
URLs\_SPEECHCOMMANDS, [569](#)  
URLs\_WIKITEXT, [570](#)  
URLs\_WIKITEXT\_TINY, [570](#)  
URLs\_WT103\_BWD, [571](#)  
URLs\_WT103\_FWD, [571](#)  
URLs\_YAHOO\_ANSWERS, [572](#)  
URLs\_YELP\_REVIEWS, [572](#)  
URLs\_YELP\_REVIEWS\_POLARITY, [573](#)

vgg11\_bn, [573](#)  
vgg13\_bn, [574](#)  
vgg16\_bn, [574](#)  
vgg19\_bn, [575](#)  
vision, [575](#)  
vleaky\_relu, [576](#)  
Voice, [576](#)

wandb, [577](#)  
WandbCallback, [578](#)  
Warp, [579](#)  
waterfall\_plot, [580](#)  
weight\_decay, [582](#)  
WeightDropout, [580](#)  
WeightedDL, [581](#)  
win\_abdoment\_soft, [583](#)  
win\_brain, [583](#)  
win\_brain\_bone, [583](#)  
win\_brain\_soft, [584](#)  
win\_liver, [584](#)  
win\_lungs, [584](#)  
win\_mediastinum, [585](#)  
win\_spine\_bone, [585](#)  
win\_spine\_soft, [585](#)  
win\_stroke, [586](#)  
win\_subdural, [586](#)

xla, [586](#)

XResNet, [587](#)  
xresnet101, [587](#)  
xresnet152, [588](#)  
xresnet18, [588](#)  
xresnet18\_deep, [589](#)  
xresnet18\_deeper, [589](#)  
xresnet34, [590](#)  
xresnet34\_deep, [590](#)  
xresnet34\_deeper, [591](#)  
xresnet50, [591](#)  
xresnet50\_deep, [592](#)  
xresnet50\_deeper, [592](#)  
xresnext101, [593](#)  
xresnext18, [593](#)  
xresnext34, [594](#)  
xresnext50, [594](#)  
xse\_resnet101, [595](#)  
xse\_resnet152, [596](#)  
xse\_resnet18, [596](#)  
xse\_resnet34, [597](#)  
xse\_resnet50, [597](#)  
xse\_resnext101, [598](#)  
xse\_resnext18, [598](#)  
xse\_resnext18\_deep, [599](#)  
xse\_resnext18\_deeper, [599](#)  
xse\_resnext34, [600](#)  
xse\_resnext34\_deep, [600](#)  
xse\_resnext34\_deeper, [601](#)  
xse\_resnext50, [601](#)  
xse\_resnext50\_deep, [602](#)  
xse\_resnext50\_deeper, [602](#)  
xsenet154, [595](#)

zoom, [603](#)  
Zoom\_, [603](#)  
zoom\_mat, [604](#)