REPRODUCIBILITY FACTOR SCORING (from 0 to 1):

Reproducibility factor	Score	Feedback	T*	A*	L*
Source-code availability and documentation	0.93	Score higher than top threshold (T). Very good.	0.8	0.54	0.28
Software environment	0.93	Score higher than top threshold (T). Very good.	0.61	0.46	0.31
Dataset availability and preprocessing	1	Score equal to top threshold (T). Very good.	1	-	0
Random seed control	1.0	Score higher than top threshold (T). Very good.	0.94	0.51	0.51
Model serialization	1	Score equal to top threshold (T). Very good.	1	-	0
Hyperparameter logging	0	Score equal to lower threshold (L). Major improvements should be made.	1	-	0
Out-of-the-box buildability	0	Score equal to lower threshold (L). Major improvements should be made.	1	-	0

> *: Thresholds computed from respective reproducible and non-reproducible sets of repositories. More information on this in the associated thesis in chapter 6.

SOURCE CODE AVAILABILITY AND DOCUMENTATION FEEDBACK

1. License scoring (score: 1.0 out of 1.0): All found licenses are open-source.
2. README overall scoring (score: 0.93 out of 1.0): See sub-ratings for more information.

- Average length scoring (score: 1.0 out of 1.0): We determined that 82 or more lines are best. We found 222.0 lines (in average) in the README file(s)
- Average accessible links scoring (score: 0.65 out of 1.0): We determined that 4 or more links are best. We found 3.0 accessible links (in average) in the README file(s)

3. Source-code pylint scoring (score: 0.63 out of 1.0): Not normalized pylint rating is 3.61 where 10.0 is best. We found that readable code is best for reproducibility and consider ratings over 5.71 as desirable.
4. Source-code-comment-ratio scoring (score: 1.0 out of 1.0): We measured a average ratio of 6.44. We consider a ratio below 8.73 as best. But more comments are fine too (which would make the ratio smaller). Well documented code is important.

> Source-code availability and documentation is calculated from the above identifiers. Since these have a different influence on the overall result, they are weighted as follows:

- License weight: 0.3
- Readme weight: 0.5
- Sub-readme: Length weight: 0.8
- Sub-readme: Average accessible links weight: 0.2
- Pylint rating weight: 0.1
- Code-comment-ratio weight: 0.1

SOFTWARE ENVIRONMENT FEEDBACK

1. Libraries used in the source-code and mentioned in config file scoring (score: 0.53 out of 0.6): We found that 87.5% of the used libraries are defined in the config file(s). We expect that 100% of the in the source-code used relevant libraries are defined in the config file(s).
2. Strictly defined libraries in config file(s) scoring (score: 0.2 out of 0.2): We expect that 100% of the defined libraries in the config file(s) are strictly (==) defined. We found 31 libraries in the config file(s). From these 100.0% were strictly specified.

- Public available libraries in source code file(s) scoring (score: 0.2 out of 0.2: We expect all of the not local modules or standard python libraries to be publicly available. We found that 100.0% are publicly available. We tested if the used library imports are accessible on https://pypi.org. If the score is not 0.2 (=100%): Please try avoiding the use of not public libraries as third parties may not be able to use your repository. Please note: It is also possible that, if the score is not the maxima, all libraries are publicly available, but we could not find a match. If you are unsure please recheck manually.

> Software environment is calculated from the above identifiers. Since these have a different influence on the overall result, they are weighted as follows:

- Source-code imports in config file weight: 0.6
- Strict dependency declarations in config file weight: 0.2
- Public libs in source code weight: 0.2

> Important note: For our analysis we exclude python standard libraries (see: https://docs.python.org/3/library/) as well as local file imports. We also eliminate duplicates (if one import occurs in multiple files we count it as one).

DATASET AVAILABILITY AND PREPROCESSING FEEDBACK

Dataset availability and preprocessing scoring (score: 1.0 out of 1.0): Dataset file candidate(s) were mentioned in the source code. This is best practice, because one should always provide a dataset (if possible) in the repository in order for others to reproduce your repository with the same input data.

> Important note: Dataset preprocessing detection is currently not implemented. But: If you preprocessed the dataset in any way please make sure to include either the final dataset or files to reproduce the steps taken.

RANDOM SEED FEEDBACK

Random seed lines with fixed seed scoring (score: 1.0 out of 1.0): We found that 100.0% of the 42 found random seed declaration lines had a fixed seed. If the value is not 100% make sure to fix the random seeds.

MODEL SERIALIZATION FEEDBACK

Model serialization scoring (score: 1.0 out of 1.0): Model serialization artifacts found. Model serialization helps in the field of machine learning, where incremental improvements should be documented in order for others to understand the steps taken. We look for one of the following: a) folders named ".dvc", b) files with the extensions ".dvc", ".h5", ".pkl" or ".model" c) one of the following keywords in the source code: "torch.save()", "pickle.dump" or "joblib".

HYPERPARAMETER LOGGING FEEDBACK

Hyperparameter logging scoring (score: 0 out of 1.0): No hyperparameter logging indicators found. We look out for imports of the following libraries in the source code: "wandb", "neptune", "mlflow" and "sacred". These libraries enable the logging of these parameters in order to document them. Also, we are looking for method calls of these libraries regarding logging. Logging changes of the hyper-parameters helps in the field of machine learning, where incremental improvements should be documented in order for others to understand the steps taken.

OUT-OF-THE-BOX BUILDABILITY FEEDBACK

Out-of-the-box buildability scoring (score: 0 out of 1.0): Tested building the repository with BinderHub resulted in an error. Repository is not buildable with BinderHub. Please try fixing this, as it greatly helps reproducing the found results. You can use the free and publicly available infrastructure accessible under https://www.mybinder.org to test. If you have included a Dockerfile in your repository it may not be compatible with BinderHub. Check: https://mybinder.readthedocs.io/en/latest/tutorials/dockerfile.html