1. **TEAM INFORMATION**

Name: Loan Vulliard

Email: [lvulliard@cemm.oeaw.ac.at](mailto:spxuw@channing.harvard.edu)

Affiliation:

CeMM Research Center for Molecular Medicine of the Austrian Academy of Sciences, Vienna, Austria.

Department of Biochemistry and Cell Biology, Max Perutz Labs, University of Vienna, Vienna, Austria

Contribution: Write the code; Analyze the results

1. **CODE DESCRIPTION/USAGE**

My source code includes two files written with Python 3.6.9.

1. crossvalidation.py

* **Description**: This code is used to calculate the performance metrics with 10-fold cross-validation from an edge list.
* **Usage**: The constants `DATASET\_FILE` and `SCORE\_FILE\_NAME` need to be set to the input and output file locations on your system. The script can then be called from the command line ("./crossvalidation.py", assuming the user has execution right for this file).
* **Hyperparameters**: None.

1. prediction.py

* **Description**: This code is used to predict the top-500 new PPIs and calculate their scores for the human interactome (HuRI).
* **Usage**: The constants `DATASET\_FILE` and `SCORE\_FILE\_NAME` need to be set to the input and output file locations on your system. The script can then be called from the command line ("./prediction.py", assuming the user has execution right for this file).
* **Hyperparameters**: None.

1. **COMPUTING ENVIRONMENT**

Scripts were run on the computing cluster of the Vienna BioCenter, asking for the following resources:

1 Tesla V100 GPU

15 CPU cores

64 GB of memory

The same code can be used with CPUs only without any modification.

1. **EXTERNAL PACKAGES/LIBRARIES**

The computing environment is based on the Docker image `tensorflow/tensorflow:latest-gpu`, pulled and run with Singularity. The libraries `pandas`, `sklearn`, `networkx` and `node2vec` were added using pip.

This resulted in the following environment:

absl-py==0.11.0

asn1crypto==0.24.0

astunparse==1.6.3

cachetools==4.2.0

certifi==2020.12.5

chardet==4.0.0

cryptography==2.1.4

decorator==4.4.2

flatbuffers==1.12

gast==0.3.3

gensim==3.8.3

google-auth==1.24.0

google-auth-oauthlib==0.4.2

google-pasta==0.2.0

grpcio==1.32.0

h5py==2.10.0

idna==2.6

importlib-metadata==3.4.0

joblib==1.0.0

Keras-Preprocessing==1.1.2

keyring==10.6.0

keyrings.alt==3.0

Markdown==3.3.3

networkx==2.5

node2vec==0.4.1

numpy==1.19.5

oauthlib==3.1.0

opt-einsum==3.3.0

pandas==1.1.5

protobuf==3.14.0

pyasn1==0.4.8

pyasn1-modules==0.2.8

pycrypto==2.6.1

pygobject==3.26.1

python-apt==1.6.5+ubuntu0.5

python-dateutil==2.8.1

pytz==2020.5

pyxdg==0.25

requests==2.25.1

requests-oauthlib==1.3.0

rsa==4.7

scikit-learn==0.24.1

scipy==1.5.4

SecretStorage==2.3.1

six==1.15.0

sklearn==0.0

smart-open==4.1.2

tensorboard==2.4.1

tensorboard-plugin-wit==1.8.0

tensorflow-estimator==2.4.0

tensorflow-gpu==2.4.1

termcolor==1.1.0

threadpoolctl==2.1.0

tqdm==4.56.0

typing-extensions==3.7.4.3

urllib3==1.26.2

Werkzeug==1.0.1

wrapt==1.12.1

zipp==3.4.0

1. **ADDITIONAL DATASET USED IN THE METHOD**

N/A

1. **METHOD DESCRIPTION**

The prediction implemented is based on the node and edge embeddings computed by the *node2vec* algorithm [1], integrated with a deep neural network designed to be invariant to the order of the node features provided.

First, the set of all edges (protein-protein interactions) provided was split into 10 subsets for cross-validation, and for each cross-validation fold, a set of non-edges (pairs of nodes absent from the original edge list) was generated both for the training and testing sets using balanced sampling as previously described and studied [2,3]. This non-edge sets were forced not to overlap to avoid a misestimation of the model performance due to non-edges used in the test set being part of the training set.

Using the *node2vec* package, I obtained a node embedding of dimension 500 for the network formed by the training set edges. I then retrieved the corresponding edge embedding using the Hadamard product to learn the edge features. These features were fed to a neural network, designed to accept three inputs of length 500, corresponding to the embedding features for the edge () and each of the corresponding nodes ( and respectively). The node features were concatenated in the order provided () as well as in the permuted order () and processed by a shared inner model linking the 1000-dimensional input to a dense layer with 200 nodes, using self-normalizing linear units (SeLu) as activation function [4]. Following an alpha dropout layer at a rate of 0.1, a dense layer of 40 nodes with linear activation formed the output of the inner model. The output for both and were integrated by max pooling, therefore ensuring that the model predictions are invariant to the order in which the node features are provided. An independent model, accepting a 500-dimensional input but of similar architecture as the node inner model, was used to reduce the edge embedding to 40 dimensions. These 40 features were concatenated to the 40 features of the node max pooling step and fed to one more dense SeLu layer with 20 nodes linked to an alpha dropout layer and finally to an output layer with a single node with a sigmoid activation function. The full model therefore maps the node and edge embedding values to a single value between 0 and 1, used as a confidence score for all potential novel protein-protein interaction. The model was trained to minimize binary cross-entropy using the Adam optimizer for stochastic gradient descent. I observed that 100 epochs were enough to reach convergence of the model regardless of the input network.

Finally, to estimate the performance of the model, I computed the confidence score for all edges and non-edges in the testing set, and obtained the corresponding AUROC, AUPRC and NDCG score using their *sklearn* implementation. Moreover, the model was run on all candidate edges (pairs of nodes in the training set that were not part of the train edges) and the fraction of the 500 candidate edges with the highest score being part of the test edges formed the p500 metric.

For the prediction, a similar embedding and neural network were trained directly on all human protein-protein interactions provided and a balanced set of non-edges of equal size. A confidence score was obtained for all candidate edges. As more than 500 edges were given a score of 1, we randomly shuffled these top candidates to obtain a final list of potential novel edges.

**Reference**

[1] Grover A, Leskovec J. node2vec. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, NY, USA: ACM, 2016: 855–64.

[2] Yu J, Guo M, Needham CJ, Huang Y, Cai L, Westhead DR. Simple sequence-based kernels do not predict protein-protein interactions. *Bioinformatics* 2010; **26**: 2610–4.

[3] Park Y, Marcotte EM. Revisiting the negative example sampling problem for predicting protein-protein interactions. *Bioinformatics* 2011; **27**: 3024–8.

[4] Klambauer G, Unterthiner T, Mayr A, Hochreiter S. Self-Normalizing Neural Networks. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. Red Hook, NY, USA: Curran Associates Inc., 2017: 972–981.