Graph Neural Networking challenge 2021:

Instructions for the evaluation phase

As described in the challenge website (https://bnn.upc.edu/challenge/gnnet2021/), the evaluation will be based on the Mean Absolute Percentage Error (MAPE) metric, computed on all the path delay predictions in the test dataset:

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{\hat{y}_i - y_i}{y_i}\right|$$

Evaluation platform and test dataset

We will release the final test dataset on September 15th at 00:00:00 (GMT+2). This dataset will contain 1,560 samples of networks considerably larger (51-300 nodes) than those of the training dataset (25-50 nodes). Particularly, **the samples of the test dataset will follow similar distributions to the ones of the validation dataset** (*i.e.*, similar topology sizes, traffic ranges, link capacities, path lengths, delays, *etc*). This dataset will not contain performance-related labels at the path or queue levels (*e.g.*, paths' delay, jitter, loss, queues' occupancy). In general, **for those using the DataNet API, no data under the *'performance_matrix'* and *'port_stats'* data structures will be included**. Note that for compatibility with the DataNet API, samples in the test dataset will return values for these data structures, but they will be constants without any particular meaning (*e.g.*, -1 for delay predictions), and they should not be used as input of the solutions.

For the evaluation, each team needs to produce labels for all the samples in the test dataset with its neural network-based solution. To do this, participants should generate a TXT file (compressed in ZIP) with the output labels and upload it to a platform we provide (see more details in Section "Prepare your submissions").

Please, access the evaluation platform at this link:
https://challenge2021.bnn.upc.edu/evaluation

If it is the first time you access the platform, you will need to create an account for your team. To do this, you should click on "Don't have an account?". **Please note that you can only register with the email address you entered as "Contact e-mail" when registering for the challenge.** Otherwise, you will get an error and the account will not be created. Likewise, only one account per team can be created.

Once an account is created, you will be able to log in using the username and password specified in the registration form. **Please, remember that despite you can register right now, the evaluation phase will be open from Sept 16th at 00:00:00 (GMT+2) to Sept 30th at 23:59:59 (GMT+2).** During this period, you will be able to submit your files there. Each time you submit a file, the platform will automatically compute the resulting MAPE score and will display it. Likewise, **a link to download the test dataset will be provided in this platform on Sep. 15th at 00:00:00 GMT+2 (one day before the evaluation phase starts).**

Prepare your submissions

Our evaluation platform only accepts ZIP files. Otherwise, you will get an error. To prepare your submission, we provide a Python script for the baseline implementation of RouteNet in Tensorflow [here] and for the implementation in IGNNITION [here], that include all the needed processing pipeline to generate the final submission files. These scripts read the samples from the test dataset, generate the delay labels with the RouteNet model, and finally creates the submission file (a TXT file compressed in ZIP). If your solution is based on any of the reference RouteNet implementations (TensorFlow or IGNNITION) you should be able to easily adapt any of these scripts to produce the output labels with your solution. To do this, please read carefully the comments within the scripts.

To run the script, you can simply execute it with the following command:

 # python generate_submission.py

More in detail, the script reads the **config file ("config.ini" in TensorFlow; "train_options.yaml" in IGNNITION**), creates and loads the model, reads the test dataset directory in a given order (more details below), makes the per-path delay predictions, saves the results in a file (defined by the "FILENAME" variable), and finally checks that the generated file has the expected format. Note that the script loads the trained model and the test datasets according to the paths defined in the config file (*i.e.*, *config.ini* in TensorFlow, or *train_options.yaml* in IGNNITION). In the TensorFlow implementation, you need to set the "logs" and "test" variables within the *config.ini* file respectively to point to the path that contains the trained model and the test dataset. In IGNNITION, you should set the "load_model_path" and "predict_dataset" variables in the *train_options.yaml* file to respectively load the trained model and test dataset.

Please, consider the three following points:

- **If your model is not based on the provided implementations of RouteNet**, you can still use the *generate_submission.py* script [here] and change the lines where the data is loaded and predictions are generated. This is referred in the script as the "Predicting block". Please, find more details in Appendix A on how to order the final delay labels in submission files.

- In order to work properly, the script only accepts a simple filename, without the extension of the submission file or the path:
  i.e., "FILENAME = submission_file", instead of "FILENAME = submission_file.zip", or "FILENAME = ./output/submission_file"

- In the TensorFlow model, if you have applied any normalization to the input of the model, you should implement it in the transformation(x, y) function. Please, also remember to denormalize/post-process the predicted values, and note that delay predictions **must have a maximum of 6 decimal digits** (this is already considered in the script)**.** You can do it adding some lines below the comment:
  "# If you need to denormalize something do it here".

Before the start of the evaluation phase, we recommend adapt the *generate_submission.py* script to your solution and validate that the format of the generated submission files is correct. To do this, we provide a toy dataset [here]. In order to validate the format, we provide in the "/code/utils" directory a file that indicates the number of paths each sample should have. Note that you need to change the "PATHS_PER_SAMPLE" variable in the *generate_submission.py* script to point to the corresponding file. Specifically:

- if you are checking the toy dataset:
  PATHS_PER_SAMPLE = *'./utils/paths_per_sample_toy_dataset.txt'*
- **if you are checking the final test dataset:**
  PATHS_PER_SAMPLE = *'./utils/paths_per_sample_test_dataset.txt'*

If the submission file has the expected format, you should see a message like this at the end of the execution:

"Congratulations! The submission file has passed all the tests! You can now submit it to the evaluation platform."

In general, it should be simpler to use directly the previous script and adapt it to your solution. However, if you are not sure on how to adapt the script to produce the submission files with your solution, we provide a detailed description of the expected format of the TXT file (compressed in ZIP) in Appendix A.

Note that all the script execution (generating predictions and checking the format) can be long. As a reference, it took for us approximately 40 min. in commodity hardware.

Rules and other evaluation details

Before you prepare your submissions, please double-check the rules on the challenge website (https://bnn.upc.edu/challenge/gnnet2021 section "Rules") to ensure your solutions comply with all of them. N**ote that each team can submit a maximum of 5 solutions (i.e., submission files) per day and, in total, teams can make up to 20 submissions during the whole evaluation phase**. Also**, solutions must be exclusively trained with samples of the training dataset we provide, or directly derived from them (*e.g.*, data augmentation).** It is not allowed to use additional data from other datasets like the validation dataset we provided or synthetically-generated data (*e.g.*, from network simulators). We will check it after the evaluation phase, as we will reproduce the training and evaluation of top-5 solutions. In this context**, please note that solutions must be sufficiently prepared to easily replicate the training/evaluation on our servers**.

During the evaluation phase, you will have access to an anonymized ranking with the 5 best scores at the time. This may help assess in real time how good are your results compared to other teams. **Each team will be considered for its best score, regardless of whether it is the last submission or not.** Also, there will be a record of all the submissions made by the team and the scores obtained in each one of them. If you have any doubt about the submissions registered by your team on our platform, you can check this record at the home page of the evaluation platform after logging in.

After the evaluation phase (Sep. 16th-Sep. 30th), we will publish a provisional ranking of all the teams (Oct 1st), and then top-5 solutions will have to send their code and documentation describing the solution and how to reproduce the training (Oct 15th). After checking that these solutions comply with all the rules, the winners (top 3) will be officially announced on Oct 31st and will be considered for the Grand Challenge Finale of the *ITU AI/ML in 5G challenge*.

Good luck! And enjoy the challenge!

Appendix A

If you are not sure if the output of your solution fits correctly the format of the output TXT file, we detail below the format this file should have:

- Each line should contain the predicted path delays for each source-destination path of the input sample, and these values (floats) **must be separated by ";"**. In total, there will be at most Nx(N-1) source-destination paths (where N is the topology size). **Please, note that it is possible that some samples contain paths with generated bandwidth (flow['AvgBw']) and packets (flow['PktsGen']) equal to zero. Predictions for these paths MUST NOT BE INCLUDED IN THE SUBMISSION FILES,** as they do not carry any traffic. Thus, for each sample the paths' delay predictions can be produced as in the following pseudo-code:

```
Line= ""
For node_src in range (N):
    For node_dst in range (N):
        If node_src != node_dst and bandwidth(src,dst) != 0 and packets(src,dst)!= 0:
            Line += str(delay[node_src, node_dst])+";"

# New sample, new line

Line +="\n"
```

  Where 'node_src' and 'node_dst' iterate from node "0" to node "N-1", according to the node ids of the matrices that the DataNet API returns for each sample.

- **Important note: The order of the samples (*i.e.*, the lines of the TXT file) matters**. To ensure they are correctly ordered you should read the test dataset with the DataNet API we provide (remember you need to use the version [here]), and **use the option "`shuffle=False`".** The test dataset will contain 3 different scenarios, each one with directories including samples of different topology size (from 50 to 300 nodes). The submission file should include predictions of these different scenarios by increasing topology size. To do so, you can use the following pseudo-code that automatically sorts them:

```
from glob import iglob

# Ensure we read the files in the correct order

directories = [d for d in iglob(TEST_PATH + '/*/*')]

# First, sort by scenario and second, by topology size

directories.sort(key=lambda f: (os.path.dirname(f), int(os.path.basename(f))))
```

Please, note that each delay prediction MUST contain a maximum of 6 decimal digits. Otherwise, the evaluation platform will truncate prediction values to this limit of digits.

In short, the TXT file must contain 1,560 lines (each one corresponding to the 1,560 samples in the final test dataset), and each line should have X values (delay predictions) corresponding to all the combinations of source-destination pairs, excluding those paths with no traffic (i.e., flow['AvgBw'] != 0 and flow['PktsGen'] != 0). Again, you can check the exact number of paths per line in the *"paths_per_sample_test_dataset.txt"* file [here].