Graph Neural Networking challenge 2020:

Instructions for the evaluation phase

As it is described in the challenge website (https://bnn.upc.edu/challenge2020), the evaluation will be based on the Mean Absolute Percentage Error (MAPE) metric. Particularly, **we will release a test dataset with 50,000 samples on September 30th** (just before the evaluation phase starts). This dataset is simulated in a different topology of 19 nodes not used in the training and validation datasets provided before, and it will not contain the output labels (i.e., the per-source-destination delays). The test dataset contains a data distribution similar to that of the validation dataset (i.e., similar ranges of traffic, delays…), so the resulting score should be similar to that obtained over the validation dataset if the model generalizes successfully. Then, each team needs to produce labels for all the samples in the dataset with its solution. To do this, you should generate a CSV file with the output labels and upload it to a platform we provide (see more details below).

You can access the evaluation platform at this link:
https://challenge2020.bnn.upc.edu/evaluation

If it is the first time you access, you will need to create an account for your team. To do this, you should click on "Create a new 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 Oct 1st at 00:00:00 (GMT+2) to Oct 15st at 23:59:59 (GMT+2).** During this period, you will be able to submit your CSV files there. Each time you submit a file, the platform will automatically compute the resulting MAPE score and will display it.

More in detail, **our platform only accepts ZIP files, so you must compress the CSV file in ZIP format before it is uploaded.** Otherwise, you will get an error. To prepare your submission, we provide a Python script [here] that includes all the necessary processing pipeline with the baseline model we provided for the challenge (RouteNet). Particularly, it reads the samples from the test dataset, generates the delay labels with the baseline model, and finally creates the output file directly in ZIP format. Thus, typically you will only need to adapt this script to produce correctly the output labels with your model. **Please, note that you are required also to update the API to read the dataset with the latest version [here] to produce correctly the CSV file**, as the previous version does not load the samples in the same order in all the machines, and we require all the samples to be in the same order to generate correctly the CSV file.

To run the script, you simply need to execute the *generate_submission.py* file provided in our GitHub repository using the command "*python generate_submission.py*". This script executes a function that takes as input the directory of the test dataset (variable "test_dataset_directory"), a filename for the output file (variable "output_filename"), and the configuration file used to define some model hyperparameters (variable "config_file_path"). Also, you need to set the "logs" variable within the *config.ini* file to point to the directory that contains the trained model. Please, consider the three following points:

- In order to work properly, the script only accepts a simple filename without the extension of the output file or the path:
  i.e., "*output_filename = submission_file*", instead of
  "output_filename=./home/dataset/submission_file.zip"
- If your model is not based on RouteNet, you can still use the script and change simply the lines where the predictions are generated. These are the lines below the "Generate predictions" code block. See also more instructions below on how to order the delay values for the output file.
- If you have applied any normalization to the input of the model, please remember to denormalize the predicted values. You can do it below the "Denormalization" code block.

You can check this script and validate the output format using the toy dataset we provide [here]. If the resulting file contains 200 lines with 342 elements each one you can assume the format is correct. Note that you will still see a warning message saying that the format is not correct. This is because the script is prepared for the final test dataset. Thus, with the final dataset you should not see this warning message.

In general, you can use directly the script and adapt it to your solution as described above. However, if you are not sure if the output of your model fits correctly the format of the CSV file, we detail below the format that this file should have:

- Each line should contain the predicted delays for each source-destination path of the input sample, and these values (floats) **must be separated by ";"**. In total, there will be 342 source-destination paths, as the network in the test dataset has 19 nodes (i.e., 19 sources x 18 destinations = 342 src-dst pairs). Thus, the delays of all the paths must be ordered according to the following pseudo-code:

```
String = ""
For node_src in range (19):
    For node_dst in range (19):
        String += str(delay[node_src, node_dst])+";"

# New sample, new line

String +="\n"
```

  Where 'node_src' and 'node_dst' iterate from node "0" to node "18", according to the node ids of the matrices that the datanetAPI returns for each sample.

- **The order of the samples (i.e., the lines of the CSV file) matters**. To ensure they are correctly ordered you should read the test dataset with the API we provide (remember you need to use the latest version [here]) and you can use the following pseudo-code:

```
tool  = datanetAPI.DatanetAPI(<pathToDataset >,shuffle=False)
it = iter(tool)
  for sample in it:
    routing = sample.get_routing_matrix()
    …
```

In short, the CSV file must contain 50,000 lines (as there will be 50,000 samples in the test dataset), and each line should have 342 values corresponding to all the combinations of source-destination paths.

Before you prepare your submissions, please double-check the rules on the challenge website (https://bnn.upc.edu/challenge2020 section "Rules") to ensure your solutions comply with all of them. Particularly, **note that each team can submit a maximum of 5 solutions (i.e., CSV files) per day and, in total, you can make up to 20 submissions**. Also**, the solutions must be exclusively trained with samples of the training dataset we provide.** It is not allowed to use additional data from other datasets like the validation dataset we provided or synthetically-generated data. We will check it after the evaluation phase, as we will reproduce the training of the top 5 solutions to check that they comply with this rule. In this context**, please note that the solutions must also be sufficiently prepared to replicate the training on our servers**. Note also that solutions must not use as input any parameter related to delay, jitter, or loss. These are the values under the '*performance_matrix*' object provided by datanetAPI. Particularly, in the test dataset all these values are set to "-1", so if the API is used to extract these values it will retrieve this default value.

During the evaluation phase, you will have access to an anonymized ranking with the 5 best scores at the time. This may help assess 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 in our platform, you can check this record at the home page after logging in.

After the evaluation phase, we will publish a provisional ranking of all the teams (Oct 16th), and then top 5 solutions will have to send their code and documentation describing the solution and how to reproduce the training (Oct 31st). After checking that these solutions comply with all the rules, the winners (top 3) will be officially announced on Nov 20th, and will be considered for the global round of the *ITU AI/ML in 5G challenge*.