Instructions and Guidelines

1. Please submit your work before the next interview round via github repo or a zip of the repo. Add a README.md file with precise instructions on how to setup and recreate your results
2. Prepare a **few minimal slides** for the open-ended descriptive portions of the task before the next round
3. Be specific with details of implementation and analysis. Make necessary assumptions as applicable
4. You are encouraged to use and modify open-source implementations
5. Use your discretion or intuition where needed. There is no single best answer
6. Choose between Task-1 **OR** Task-2
7. Tip: Use google colab for free GPUs

Task 1 – Thermal AI-Digital Twins / Surrogate Models

Siemens Energy AI Lab

Context

All across the world, system operators are confronting challenges with congested power grids. Maintaining reliability and affordability as more loads come online from renewable resources, is a very complex and daunting task. This makes the task of operating the power grids increasingly difficult, if not impossible, to do with existing ways of working.

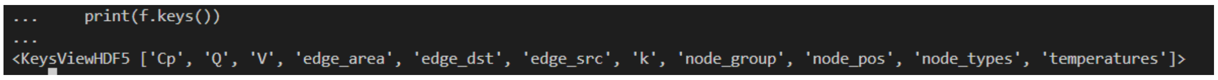
One way to achieve this is by overloading assets (i.e. *dynamic asset rating*) and increasing the capacity of the network. At a substation level, this could be for assets like Transformers or Gas-Insulated Switchgears (GIS - shown below), depending on the operating conditions like ambient temperature and load.

In this task, you will be developing surrogate models using CFD simulation data for **one module of the GIS,** under varying input conditions. We are interested in -

1. Predicting the thermal behavior i.e. spatio-temporal temperatures across the module
2. Predicting the hotspot (hottest spot) over time

Data Explanation

1. GIS dataset is a set of hdf5 file. Each file is a time-series graph (120 timesteps corresponding to ~10h in real-time; 1739 graph nodes) derived from a high-res mesh and contains the solution temperature field from CFD simulations
2. Inputs to the model are current (***I_curr***) and ambient temperature (***T_amb***) which are fixed per data file. There are 10 files in total
3. The relevant hdf5 keys in each file are given below (ignore the rest)
    a. ***Cp*** and ***k*** are node-specific scalar properties like thermal conductivity, etc. of shape (1739,)
    b. ***edge_src*** & ***edge_dst*** – Edge connectivity information with shape ()
    c. ***node_types*** – node type index value
    d. **node_pos** - position of nodes in 3D space
    e. **temperatures** – The temperature of each node across time. Our desired output



```
...     print(f.keys())
...
<KeysViewHDF5 ['Cp', 'Q', 'V', 'edge_area', 'edge_dst', 'edge_src', 'k', 'node_group', 'node_pos', 'node_types', 'temperatures']>
```

Task Instructions

1. Download the linked CFD simulation datasets that corresponds to
    a. Transient simulations of Gas-Insulated Switchgear module under varying current and ambient temperatures (***I_curr***, ***T_amb***)
    b. Split into 80% train, 20% test data

Write high-quality well-structured & documented code, using python best-practices for -

2. **[WITH CODE]**
   How would you analyze the data across spatial and temporal dimensions? For example - Visualize/Plot each graph dataset based on ***node_types*** and ***temperature*** segmentations for each file. Use tools like PyVista or similar tools

3. **[WITH CODE]** Implement a ML model that can learn the physics of heat transfer and estimate the temperature across the geometry for an unseen par of inputs (***I_curr***, ***T_amb***). The model should predict the temp. at each time-step across at each node
    a. Implement Transolver [here], FNO or MeshGraphNet models to predict steady-state temperature i.e. last time-step
    b. **Bonus**: How would you adapt or modify the architecture for full-transient behavior? i.e. predicting temp. across time

4. **[1 SLIDE]** Devise suitable metrics (at least 2) with reasons for evaluating the model
5. **[1 SLIDE]** Identify ways to scale the model training assuming data and compute are available, for much larger geometries (ex: complete GIS structure)

Task 2 – Few-shot synthetic data generation

Asset maintenance and inspection is a huge part of operating power lines and substations. Doing it at scale over 1000s of kms of power lines is a challenging task. But early identification of visible degradation, faults or other defects can be a huge cost-saving. However, we have only a handful instances of anomalous image data for each ‘defect type’. One way to train our models is by using synthetic data. In this task, we will use a publicly available dataset

Download the DefectSpectrum/Defect_Spectrum · Datasets at Hugging Face dataset that contains images of damaged industrial products along with segmentation masks and captions. Implement high-quality well-structured & documented code, using python best-practices, for the following –

1. **[WITH CODE]** A performant & efficient dataloader class that is optimized for common image pre-processing. Describe how you’d modify and further optimize for multi-GPU implementations?
2. Choose a suitable pre-trained (diffusion) model from HuggingFace for few-shot image generation based on your available compute. Select appropriate hyper-params, data config (resolution, etc) and fine-tuning methods
3. Select one damaged product class (ex: zipper, pill) and **X samples (ex: 5,10,50,100)** from each *damage-type*
4. **[WITH CODE]** Implement a model fine-tuning for generating new class-conditional images of that damage type
5. **[SLIDE]** How can you utilize the segmentation masks for this task?
6. **[Bonus]** Textual Inversion: Learn a new text embedding for token (ex: sks_scratch) that represents the specific *damage type*. Can you generate images with a prompt using the learned embedding (ex: "a photo of a pill with vertical sks_scratch ")?
7. Devise automated, quantitative metrics and other non-qualitative methods for evaluating the quality of images generated