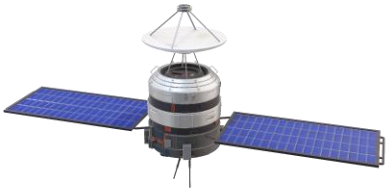
Potential improvements

1. Task 1 🌄 Text Mountain Classification

1) Dataset creation.
   Now, the synthetic dataset is created for the NER model training, but the real data is more valuable for development. The model's performance and accuracy are dependent on the quality and diversity of data.
   In this project we use only 25 mountain names and 10 templates, which leads to model overfitting. Thus, model will learn specific sentence structures and will work worse with different text.
   There are many ways to handle that problem. First, it's possible to add more unique templates, with complex grammatical structures, affirmations, negations, and questions. Second, there may be some negative samples without mountain names, their tokens will all be "O". Third, it's necessary to add templates with 2 or more mountain names.
   However, real datasets will be the best for the algorithm. There is a chance to search the internet and find texts that mention different mountains and use them in training.
2) Requirements.
   It's recommended to specify versions of packages. In future, it can lead to reproducibility issues, since pip manager installs a new version of a package.
3) Model arguments.
   It's better to spend more time on hyperparameters configuration, as there are better ways to achieve higher performance and accuracy.
4) General functionality. As a result, the algorithm works correctly and extract all mountain names in different sentences. On the contrary, there are probably some logical mistakes and minor bugs, which can result in unpredictable behavior in specific situations.
   There are better solutions for that task in terms of the code complexity.

2. 🛰️🌌 Sentinel-2 image matching. LoFTR. Detector-Free Local Feature Matching with Transformers.



1) **Dataset preparation.**

Now, we need to download a complete dataset from Kaggle, which weighs more then 30 GB, that's insane. It's better to copy all jp2 files and upload them to the cloud.

Moreover, we match only 2 images at once. Probably, it's better to combine many ones into batches and work with them.

2) Image size.

In this implementation we reduce the image size by 10 times, so we lose quality and many small details, which are critical.

However, if we reduce the size only by 2 times (5490 by 5490), it will require significantly more computing resources and time. Thus, balance should be found.

In addition, it's possible to split the images into smaller images and analyze them accordingly.

3) Post-processing of results.

We have static satellite images, so maybe it's better to use findHomography function to determine inliers by finding a homography matrix.

4) General functionality. Overall, there is possible to get rid of code repetitions, and optimize some functions.