Possible Enhancements

Results

Metrics:
- I used the F1 score as the primary evaluation metric due to class imbalance, achieving a score of 0.912 during evaluation.

Demo:
- The model effectively distinguishes between mountain names and other entities, even in challenging cases where names overlap with non-mountain references.

Areas for Improvement

This section is divided into two parts: Dataset and Model.

Dataset

1. I utilized Llama 3.1 for synthetic dataset generation, which produced similar sentences for each mountain entity. This resulted from using the same prompt across multiple mountains. While effective for a small number of examples, this approach became repetitive, and even with a temperature of 0.9, diversity remained limited. A potential improvement would be to develop a system that first generates diverse prompts, which can then be used to generate sentences in a more varied and less deterministic manner.

2. I collected the samples with mountains from a few-nerd dataset and concatenated them with generated synthetic samples with mountains. Then balanced this dataset with sentences without mountains from both wnut16 and few-nerd with a 50/50 ratio. The exact ration may be considered as a thing we need to explore in detail as it directly influences our model predictive capabilities.
3. Besides experimenting with ratios we also could investigate the examples our model treats bad and augment our dataset with such ones. In instance we could experiment with different preprocessing

of special symbols and sentences at all. For now I found using the original case and punctuation a reasonable approach as they also could impact pattern recognition.
4. As always, a larger dataset is the best way to have a better model across the board, especially in transformer usage scenarios. It also would work well to handle overfitting.

Model

1. I did not conduct a large scale model selection due to time limits and stopped on using the bert-base-ner fine-tune for NER on CoNLL-2003. Even there, we could try to use a larger version of it or explore other architectures.
2. Adjusting hyperparameters could lead to significant performance gains. For example, tuning the class weights parameter could address class imbalance, while adjusting weight_decay could help mitigate overfitting. Furthermore, fine-tuning the number of epochs, using learning rate schedulers, and optimizing other hyperparameters can have a meaningful impact on the model's predictive performance.