Assignment on Entity Set Expansion using SetExpander

SetExpander is a corpus-based system for expanding a set of seed terms into a richer set of terms that belong to the same semantic class [1]. It uses an iterative algorithm where the user can set an initial set of seed terms, expand them, validate and update the seed terms using the expanded terms and re-expand them.

We are developing OpenKG: an end-to-end framework for knowledge graph generation from unstructured data. An essential component of the framework is named entity recognition (NER), where users can use a pretrained model for extracting entities or train their model. Since training supervised NER model from scratch requires a large amount of labeled data, we will incorporate SetExpander to allow users to expand entities using a small number of seed entities. The implementation of SetExpand is available in IntelLabs.[1] You are required to incorporate it in the OpenKG framework.

Before starting with implementation, read the paper [1], SetExpander documentation[2] and implementation [3].

You are required to complete the following for the assignment:

1. Run *test_ner.py* located in *openkg/entity_extraction* directory from the *src* directory. If you use PyCharm (recommended), make *src* directory as "Sources Root".
2. Implement the *SetExpander* class in textitopenkg/entity_extraction/set_expander.py and add its unit tests in *test_set_expander* method.
   (a) First, complete the inference method *get_entities*. You can use pretrained models provided in SetExpander for this (refer to Inference-1.Running a python script in the SetExpander documentation). The user should be able to provide model path and other required parameters in *config* dictionary during initialization.
   (b) Second, implement the training method (*train*). The user can provide the corpus and other required parameters supported by SetExpander (refer to Training section in the SetExpander documentation).
3. In current SetExpand implementation, the user can provide a parameter *topn*, which determines how many maximum entities will be returned. Add another parameter, *mininum_similarity_score*. You need to implement it so that the algorithm will only return entities that have similarity score of at least *mininum_similarity_score* with any of the seed entities.
4. The iterative process of the current SetExpander implementation involves manually updating the entities for the next iteration and rerunning the algorithm. Automate this process so the user can provide a parameter *num_iteration*, and the process will be repeated as many times.

References

[1] Mamou, J., Pereg, O., Wasserblat, M., Eirew, A., Green, Y., Guskin, S., Izsak, P., and Korat, D. Term set expansion based nlp architect by intel ai lab. *arXiv preprint arXiv:1808.08953* (2018).

[1] https://github.com/IntelLabs/nlp-architect
[2] https://intellabs.github.io/nlp-architect/term_set_expansion.html
[3] https://github.com/IntelLabs/nlp-architect/blob/master/solutions/set_expansion/set_expand.py