Sequeval: A Framework to Assess and Benchmark Sequence-based Recommender Systems

Diego Monti[a], Enrico Palumbo[b, c, a], Giuseppe Rizzo[b] and Maurizio Morisio[a]

[a]Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Turin, Italy
[b]Istituto Superiore Mario Boella (ISMB), Via Pier Carlo Boggio 61, 10138 Turin, Italy
[c]EURECOM, Sophia Antipolis, Campus SophiaTech, 450 Route des Chappes, 06410 Biot, France

Introduction

Traditional Recommender Systems (RSs) usually do not consider the temporal dimension of user preferences when suggesting a set of items. This simplifying hypothesis may represent a limitation in domains characterized by the rapid consumption of different items one after the other. Even if some authors proposed RSs capable of considering training items as sequences [1], the idea of suggesting sequences of items instead of lists ranked by relevance is not widespread. On the other hand, this problem is quite similar to the task of generating a phrase given its initial words [2].

A Concrete Use Case

Consider, for example, the context of music streaming services. Starting from the human-curated playlists already available in the system, a sequence-based RS should be capable of creating a personalized playlist for the target user given an initial seed. The initial seed may be a song, a set of songs, or a genre.

In this work, we present `sequeval`, a Python implementation of an evaluation framework designed for comparing sequence-based RSs. This software package is freely available on a GitHub repository at `https://github.com/D2KLab/sequeval`.

Demonstration and Usage

In order to exploit the proposed framework, it is necessary to create an implementation of the abstract recommender that must be capable, given the user and the current item of the sequence, of predicting the probabilities for all the possible items of being the next one inside the recommended sequence.
For demonstrative purposes, we have included in `sequeval` a down-sampled version of the playlists dataset originally collected by Shuo Chen from the *Yes.com* website [3]. Furthermore, we have realized a web-based version of the same evaluation script in order to provide a more convenient graphical interface to perform the experiments, as shown in Figure 1.

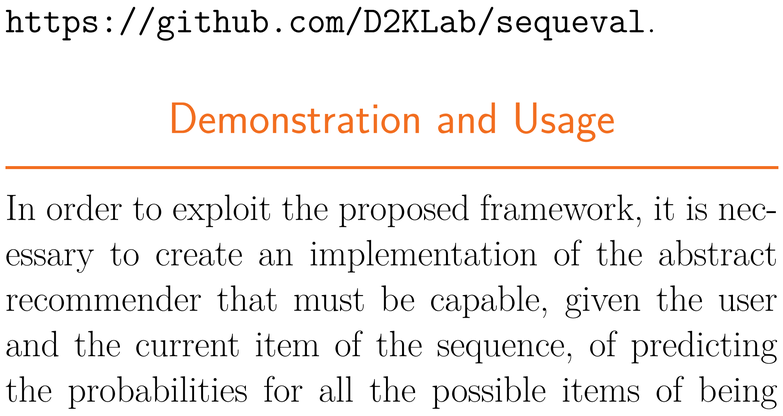

Figure 1: Screenshot of the web-based interface

System Architecture

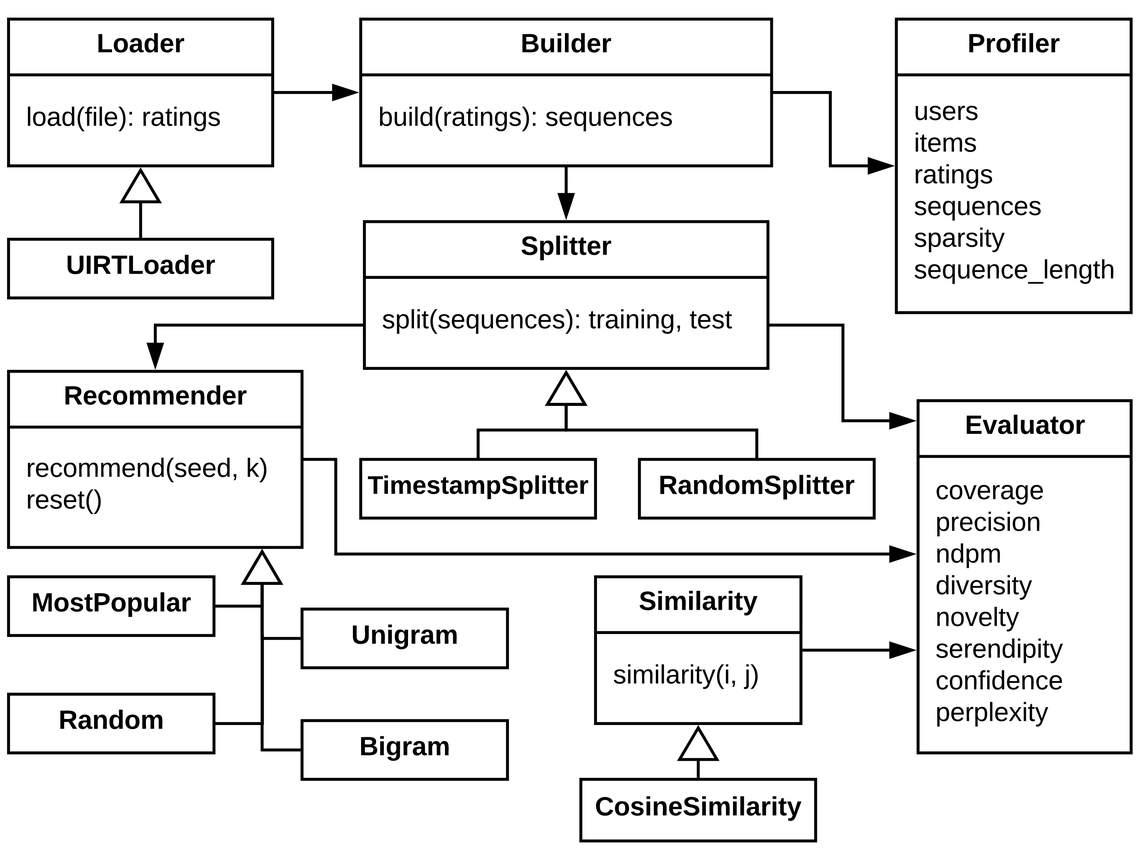

Figure 2: Simplified UML class diagram

Recommenders

We have included in `sequeval` four baseline recommenders, which represent an adaptation of traditional non-personalized baselines to the sequence-based scenario.

Most Popular The most popular recommender only considers the popularity of the items in the sequences available in the training set in order to create the recommended sequence.

Random The random recommender simply creates sequences that include random items.

Unigram The unigram recommender generates sequences that contain items sampled with a probability proportional to the number of times they were observed in the training set.

Bigram The bigram recommender estimates the 1-*st* order transition probabilities among all possible pair of items available in the training sequences.

Evaluation Metrics

In order to provide a comprehensive analysis of the recommended sequences, we have included in the *evaluator* module eight different metrics.
We decided to consider traditional metrics like coverage and precision, and also less common ones like novelty, diversity, and serendipity. We have also introduced the metric of perplexity, because it was created for evaluating sequences [4].
For each sequence in the test set, a sequence of a certain length is generated by the chosen recommender, considering the same target user and the first item of the test sequence as the initial seed. The length of the sequences is a parameter of the experiment.
In details, the metrics are:

- Coverage
- Precision
- nDPM
- Diversity
- Novelty
- Serendipity
- Confidence
- Perplexity

Conclusion and Future Work

We presented `sequeval`, a software tool for evaluating sequence-based RSs. As future work, we plan to include further datasets and more recommenders.

References

[1] Ruining He, Wang-Cheng Kang, and Julian McAuley.
Translation-based recommendation.
In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pages 161–169. ACM Press, 2017.

[2] Daniel Jurafsky and James H. Martin.
Speech and Language Processing.
Prentice Hall, 2008.

[3] Shuo Chen, Josh L. Moore, Douglas Turnbull, and Thorsten Joachims.
Playlist prediction via metric embedding.
In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 714–722. ACM Press, 2012.

[4] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin.
A neural probabilistic language model.
Journal of Machine Learning Research, 3:1137–1155, 2003.