Beyond the Repo: A Case Study on Open Source Integration with GECToR

Sanjna Kashyap, Zhaoyang Xie, Kenneth Steimel, Nitin Madnani
Educational Testing Service (ETS)

Abstract

We present a case study describing our efforts to integrate the open source GECToR code and models into our production NLP pipeline that powers many of Educational Testing Service's products and prototypes.

What is GECToR?

GECToR (Grammatical Error Correction Tag, not Rewrite) is a set of deep learning models developed by Grammarly for the task of Grammatical Error Correction or GEC. GECToR achieves state-of-the-art results for the GEC task and its inference speed is up to **10 times as fast** as that of equivalent Transformer-based sequence-to-sequence (seq2seq) GEC systems.

Issues Faced

GECToR was open sourced as part of a research publication and aligns well with the needs of the academic community. However, when it came to using it in a commercial context, we faced certain problems.

- Not under active development
- Used significantly older versions of Python, Pytorch and AllenNLP
- No versioning or packaging
- Did not fully exploit high level abstractions and APIs provided by AllenNLP

Solutions Implemented

- **Add unit and regression tests** - Checks if our modifications affect the predictions of the model.
- **Update all dependencies** - Leads to compatibility with modern infrastructure and fewer dependency conflicts.
- **AllenNLP-ify GECToR** - Facilitates better configurability and ease of use.
- **Add versioning and create conda package** - Makes installation of GECToR much easier and reproducible.

Lessons Learned

Projects should explicitly state a purpose.

Open source authors should explicitly state the purpose of their project. This allows potential adopters to estimate if the library is ready for integration.

Estimation of effort is hard but necessary.

It is crucial to perform a careful analysis of the effort involved in integrating an open source project into a production codebase. Larger projects under active maintenance will have a higher chance of fixing bugs or implementing features, and so might be easier to use, whereas smaller projects might involve more hands-on effort to prepare it for integration.

Test, test, test!

A good testing setup is critical irrespective of the eventual use case since it assures users that the code actually behaves as expected.

Always have a contingency plan.

One of the primary concerns with open source projects is the risk of abandonment by their original authors. It is important to be prepared for such scenarios and have a well-defined strategy in place for either taking on the entire maintenance of the library or transitioning to an alternate solution.

Future Work

Transition away from AllenNLP as it has been archived.

Link to paper

Link to Github