The Journal of Open Source Software

pygamma-agreement : a Python implementation of the Gamma inter-annotator agreement

Rachid Riad[1, 2] **and Hadrien Titeux**[1]

1 LSCP/ENS/CNRS/EHESS/INRIA/PSL Research University, Paris, France **2** NPI/ENS/INSERM/UPEC/PSL Research University, Créteil, France

DOI: 10.21105/joss.0XXXX

Software

- Review
- Repository
- Archive

Editor: Editor Name

Submitted: 01 January XXXX
Published: 01 January XXXX

Introduction

A great part of the current efforts in Linguistic Studies and automated speech processing algorithms goes into recording audio corpora that are ever bigger in size and ever more diverse in origins. However, an audio corpus is close to useless for many applications if it hasn't been painstakingly annotated by human annotators to produce a reference annotation, that reliably indicates events contained in the audio track (be it speech (I. McCowan & Wellner, 2005), baby noises (Cychosz, 2019), animal vocalizations (Potamitis et al., 2014), or even just plain noises (Snyder et al., 2015)). Moreover, indicating *when* something happens in the audio is half of the work: in many cases, it's also important for the human annotator to indicate *what* is the nature of the event. Indeed, many annotations are either categorical, or - in the case of speech - precise transcriptions (Serratrice, 2000) of the recorded speech. However, human annotators are suceptible to biases and errors, which raises the obvious question of the consistency and the reproducibility of their annotations. For these reasons, small parts of a corpus are usually annoted several times by different annotators, to assess the *agreement* between annotators, and thus establish a numerical measure of the difficulty of annotating this corpus.

Consequently, the Gamma (γ) Inter-Annotator Agreement Measure was proposed by (Mathet et al., 2015). This statistical measure combines both of the common agreement paradigms : unitizing (*where* are the annotations) and categorization (*what* are the annotations).

The authors of (Mathet et al., 2015) provided a Java freeware (and thus closed-source) GUI implementation. However, a lot of the work in either automated speech processing or linguistics today is done using Python or shell scripts. For this reason, we thought it would greatly benefit both communities if we could provide them with a fully open-source Python implementation of the original algorithm.

The pygamma-agreement Package

The `pygamma-agreement` package provides users with two ways to compute (in Python) the γ-agreement for a corpus. The first one is to use the simple Python API.

```
import pygamma_agreement as pa
continuum = pa.Continuum.from_csv("data/PaulAlexSuzann.csv")
dissimilarity = pa.Dissimilarity(categories=list(continuum.categories))
gamma_results = continuum.compute_gamma(dissimilarity, confidence_level=0.02)
print(f"Gamma is {gamma_results.gamma}")
```

The most important primitives from our API (the `Continuum` Figure 1 and `Alignment` Figure 2 classes) can be displayed using the `matplotlib.pyplot` backend if the user is working in a Jupyter notebook.

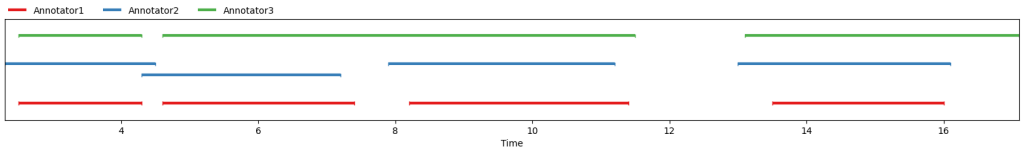

Figure 1: Displaying a Continuum in a jupyter notebook.

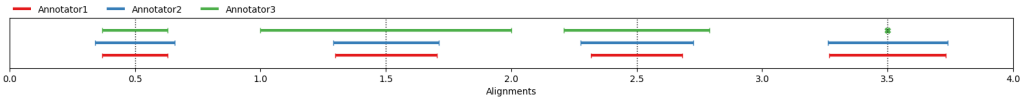

Figure 2: Displaying an Alignment in a jupyter notebook.

The second one is a command-line application that can be invoked directly from the shell, for those who prefer to use shell scripts for corpus processing:

```
pygamma-agreement corpus/*.csv --confidence_level 0.02 --output_csv results.csv
```

We support an array of commonly used annotation formats: RTTM, TextGrid, CSV and `pyannote.core.Annotation` objects.

Computing the gamma-agreement requires both array manipulation and some convex optimization. We thus used Numpy for array operations. Since some parts of the algorithm are fairly demanding, we made sure that these parts were heavily optimized using `numba` (Lam et al., 2015). The convex optimization is done using `cvxpy` (Diamond & Boyd, 2016)'s MIP-solving framework. For time-based annotations, we rely on primitives from `pyannote.core` (Bredin et al., 2020). We made sure that it is robustly tested using the widely-adopted `pytest` testing framework. We also back-tested it against the original Java implementation.

We provide a documentation as well as an example Jupyter notebook in our package's repository. Additionally, we've used and tested `pygamma-agreement` in conjunction with the development of our own custom-built annotation platform, Seshat (Titeux et al., 2020).

We've uploaded our package to the Pypi repository, thus, `pygamma-agreement` can be installed using pip.

Future Work

We've identified a small number of improvements that our package could benefit from:

- A low hanging fruit is to add the support for the "γ-cat" metric, a complement measure (Mathet, 2017) for the γ-agreement.
- The γ-agreement's theoretical framework allows for the inclusion of a sequence-based dissimilarity, based on the Levenshtein distance.
- While our implementation is already close to the fastest pure python can be, we've identified some parts of it that could benefit from `numba`'s automatic parallelization features.

Acknowledgements

We are thankful to Yann Mathet's help on understanding his work. This work is funded in part by the Agence Nationale pour la Recherche (ANR-17-EURE-0017Frontcog, ANR-10-IDEX-0001-02 PSL*, ANR-19-P3IA-0001PRAIRIE 3IA Institute) and Grants from Neuratris, from Facebook AI Research (Research Gift), Google (Faculty Research Award), Microsoft Research (Azure Credits and Grant), and Amazon Web Service (AWS Research Credits).

References

Bredin, H., Yin, R., Coria, J. M., Gelly, G., Korshunov, P., Lavechin, M., Fustes, D., Titeux, H., Bouaziz, W., & Gill, M. (2020). Pyannote.Audio: Neural building blocks for speaker diarization. *ICASSP 2020 - 2020 Ieee International Conference on Acoustics, Speech and Signal Processing (Icassp)*, 7124–7128. https://doi.org/10.1109/ICASSP40776.2020.9052974

Cychosz, C., M. (2019). *Canonical babble development in a large-scale crosslinguistic corpus*. https://doi.org/https://doi.org/10.31234/osf.io/9vzs5

Diamond, S., & Boyd, S. (2016). CVXPY: A python-embedded modeling language for convex optimization. *J. Mach. Learn. Res.*, *17*(1), 2909–2913.

I. McCowan, W. K., J. Carletta, & Wellner, P. (2005). *The ami meeting corpus*.

Lam, S. K., Pitrou, A., & Seibert, S. (2015). Numba: A llvm-based python jit compiler. *Proceedings of the Second Workshop on the Llvm Compiler Infrastructure in Hpc*. https://doi.org/10.1145/2833157.2833162

Mathet, Y. (2017). The Agreement Measure Gamma-Cat : a Complement to Gamma Focused on Categorization of a Continuum. *Computational Linguistics*, *43*(3), 661–681. https://doi.org/10.1162/COLI/_a/_00296

Mathet, Y., Widlöcher, A., & Métivier, J.-P. (2015). The unified and holistic method gamma ( ) for inter-annotator agreement measure and alignment. *Computational Linguistics*, *41*(3), 437–479. https://doi.org/10.1162/COLI_a_00227

Potamitis, I., Ntalampiras, S., Jahn, O., & Riede, K. (2014). Automatic bird sound detection in long real-field recordings: Applications and tools. *Applied Acoustics*, *80*, 1–9. https://doi.org/https://doi.org/10.1016/j.apacoust.2014.01.001

Serratrice, L. (2000). Book reviews : The childes project: Tools for analyzing talk, 3rd edition. *First Language*, *20*(60), 331–337. https://doi.org/10.1177/014272370002006006

Snyder, D., Chen, G., & Povey, D. (2015). *MUSAN: A music, speech, and noise corpus*. http://arxiv.org/abs/1510.08484

Titeux, H., Riad, R., Cao, X.-N., Hamilakis, N., Madden, K., Cristia, A., Bachoud-Lévi, A.-C., & Dupoux, E. (2020, May). Seshat: A tool for managing and verifying annotation campaigns of audio data. *LREC 2020 - 12th Language Resources and Evaluation Conference*. https://hal.archives-ouvertes.fr/hal-02496041