The Journal of Open Source Software

pygamma-agreement: Gamma γ measure for inter/intra-annotator agreement in Python

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

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.02989

Software

- Review
- Repository
- Archive

Editor: Arfon Smith

Reviewers:

- @faroit
- @apiad

Submitted: 25 November 2020
Published: 12 June 2021

Introduction

Over the last few decades, it has become easier to collect large audio recordings in naturalistic conditions and large corpora of text from the Internet. This broadens the scope of questions that can be addressed in speech and language research.

Scientists need to challenge their hypotheses and quantify the observed phenomenons of speech and language; this is why researchers add different layers of annotations on top of speech and text data. Some types of human intervention are used to reliably describe events contained in the corpus's content (e.g., Wikipedia articles, conversations, child babbling, animal vocalizations, or even just environmental sounds). These events can either be tagged at a particular point in time, or over a period of time. It is also commonplace to provide a categorical annotation or - in the case of speech - even precise transcriptions (Serratrice, 2000) for these events. Depending on the difficulty of the annotation task and the eventual expertise of the annotators, the annotations they produce can include a certain degree of interpretation. A common strategy when building annotated corpora is to have small parts of a corpus annotated by several annotators to be able quantify their consensus on that reduced subset of the corpus. If that consensus is deemed robust (i.e., agreement is high), we infer that the annotation task is well defined, less prone to interpretation, and that annotations that cover the rest of the corpus are reliable (Gwet, 2012). An objective measure of the agreement (and subsequent disagreement) between annotators is thus desirable.

Statement of Need

The Gamma (γ) Inter-Annotator Agreement Measure was proposed by Mathet et al. (2015) as a way to quantify inter-rater agreement for sequences of annotations. The γ-agreement measure solves a number of the shortcomings of other pre-existing measures. This quantification will have to satisfy some constraints: segmentation, unitizing, categorization, weighted categorization and the support for any number of annotators. They should also provide a chance-corrected value. Other measures, such as the κ (Carletta, 1996) or Krippendorff's α's (Krippendorff, 2011), have existed for some time to deal with these constraints, but cannot address all of them at once. A detailed comparison between metrics is available in Mathet et al. (2015).

To solve all of these constraints at once, the γ-agreement works in three steps: 1) a *disorder* (i.e., a cost) is computed for each potential alignments between the different annotators' units, using an annotation-dependent *dissimilarity* (akin to a distance between units). This *disorder* models the disagreement between annotators. 2) Using a convex optimization algorithm, a global alignment with the lowest total disorder is found. 3) By sampling random and synthetic

annotations from the original annotation and computing their own disorders, the original annotation's disorder value is chance-corrected to provide the final γ-agreement measure. The authors of Mathet et al. (2015) provided a Java freeware GUI implementation along with their paper and this implementation has already been used by some researchers (Da San Martino et al., 2019) to compute an inter-rater agreement measure on their annotations.

However, linguist and automated speech researchers today use analysis pipelines that are either Python or shell scripts. To this day, no open-source implementation allows for the γ-agreement to be computed in a programmatical way, and researchers that are already proficient in Python and willing to automate their work might be hindered by the graphical nature of the original Java implementation. Moreover, the original γ-agreement algorithm has several parameters that are determinant in its computation and cannot be configured as of now. For this reason, it would greatly benefit the speech and linguistic scientific community if a fully open-source Python implementation of the original algorithm was available – this is what we are making availble here. We have made sure that our implementation has several key features:

- It is comparatively as fast as the original implementation, taking about 10s to compute a high-confidence γ-agreement measure on a middle-range processor.
- The `pygamma-agreement` package is modular and users can easily extend one of the modules without the burden of rebuilding an optimized code base from scratch (e.g., Users can easily add a new dissimilarity measure).
- Our code allows for fine-grained constructions of an annotation Continuum and an advanced configurability of the γ-agreement's different parameters.
- We also made sure the interpretability of the input (i.e., the annotation continuum) and the output (i.e., the alignments) data structures are straightforward, as both can be visualized in a Jupyter Notebook (see Figure 1).
- Finally, we support most of the commonly used data formats and analysis pipelines in the speech and linguistic fields.

The pygamma-agreement Package

The `pygamma-agreement` package provides users with two ways to compute the γ-agreement for a corpus of annotations. The first is to use the package's Python API.

```
import pygamma_agreement as pa
continuum = pa.Continuum.from_csv("data/PaulAlexSuzann.csv")
dissimilarity = pa.CombinedCategoricalDissimilarity(categories=list(continuum.cate
gamma_results = continuum.compute_gamma(dissimilarity, precision_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.

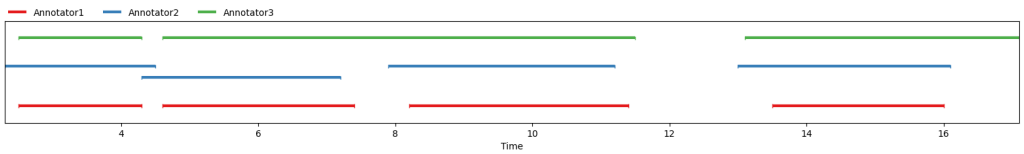

Figure 1: Displaying a Continuum in a jupyter notebook. This is a temporally accurate representation of the annotated data.

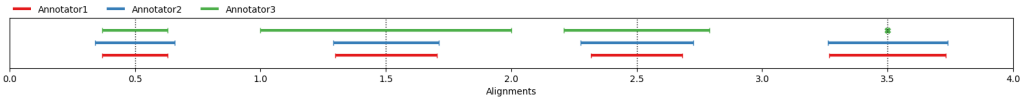

Figure 2: Displaying an Alignment in a jupyter notebook. This is a visual and schematic representation of the alignment computed between annotations of the original continuum (the order of units is respected but the annotations are scaled for visual clarity).

The second 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 a variety of commonly used annotation formats among speech researchers and linguists: RTTM, ELAN, TextGrid, CSV and `pyannote.core.Annotation` objects.

Computing the γ-agreement requires both array manipulation and the solving of multiple optimization problem formulated as Mixed-Integer Programming (MIP) problems. We thus used the *de facto* standard for all of our basic array operations, NumPy (Harris et al., 2020). Since some parts of the algorithm are fairly demanding, we made sure that these parts were heavily optimized using `numba` (Lam et al., 2015). We used `cvxpy`'s (Diamond & Boyd, 2016) MIP-solving framework to solve the optimization problem. 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 made sure that `pygamma-agreement`'s outputs matched both the theoretical values from the original paper and those of the Java freeware. Travis CI is used to run tests to ensure package quality is maintained and most of the code is type-hinted and has descriptive docstrings, both of which can be leveraged by IDEs to ease the use of the API.

We provide a user documentation as well as an example Jupyter notebook in the package's repository. Additionally, we have used and tested `pygamma-agreement` in conjunction with the development of our own custom-built annotation platform, Seshat (Titeux et al., 2020). In Table 1, we present two use cases for our implementation of the γ-agreement measure on two corpora. These two corpora, ranging from medical interviews to child recording, allowed us to evaluate the performance of the γ-agreement on a wide panel of annotation types.

Table 1: γ Inter-rater agreement for clinical interviews (16 samples) and child-centered day-long recordings (20 samples).

Corpus	Annotation	# Classes	Mean of γ
Clinical Interviews	Turn-Takings	3	0.64
Clinical Interviews	Utterances	1	0.61
Child Recordings	Speech Activity	1	0.46
Child Recordings	Child/Adult-directed speech	2	0.27

The package is published in PyPi, thus, `pygamma-agreement` can be installed using pip.

Future Work

This implementation of the γ-agreement opens the path for a number of potential extensions:

- An obvious improvement is to add support for the "γ-cat" metric, a complement measure

(Mathet, 2017) for the γ-agreement.

- The γ-agreement's theoretical framework allows for some useful improvements such as:
  - The implementation of new dissimilarities, such as a sequence-based dissimilarity (based on the Levenshtein distance), an ordinal dissimilarity (for ordered sets of categories) and a scalar dissimilarity.
  - For categorical annotations, the support for an undefined set of categories, with annotators using different sets of categories. This would be solved using an adapted implementation of the Hungarian Algorithm. This could be useful in unconstrained diarization annotation tasks.
  - More hypothetically, the possibility to have so-called "soft alignments," where a unit has weighted alignments with other units in its continuum.

Acknowledgements

We are thankful to Yann Mathet for his help on understanding his work on the γ-agreement. We also thank Xuan-Nga Cao, Anne-Catherine Bachoux-Lévy and Emmanuel Dupoux for their advice, as well as Julien Karadayi for helpful discussions and feedback. 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

Carletta, J. (1996). Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, *22*(2), 249–254. https://www.aclweb.org/anthology/J96-2004

Da San Martino, G., Yu, S., Barrón-Cedeño, A., Petrov, R., & Nakov, P. (2019). Fine-grained analysis of propaganda in news article. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 5636–5646. https://doi.org/10.18653/v1/D19-1565

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

Gwet, K. Li. (2012). *Handbook of inter-rater reliability.* 6–7.

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., R'ıo, J. F. del, Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Krippendorff, K. (2011). *Computing krippendorff's alpha-reliability.* https://repository.upenn.edu/asc_papers/43/

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 (γ) for inter-annotator agreement measure and alignment. *Computational Linguistics*, *41*(3), 437–479. https://doi.org/10.1162/COLI_a_00227

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

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