Improving NER Research Workflows with SeqScore

Constantine Lignos[†] and **Maya Kruse**[*] and **Andrew Rueda**[*]
Michtom School of Computer Science
Brandeis University
{lignos,mayakruse,andrewrueda}@brandeis.edu

Abstract

We describe the features of SeqScore, an MIT-licensed Python toolkit for working with named entity recognition (NER) data. While SeqScore began as a tool for NER scoring, it has been expanded to help with the full lifecycle of working with NER data: validating annotation, providing at-a-glance and detailed summaries of the data, modifying annotation to support experiments, scoring system output, and aiding with error analysis. SeqScore is released via PyPI and development occurs on GitHub.

1 Introduction

While much attention in language technology development is focused on the creation of better models and datasets, it is essential to also have tools for understanding the output of those models and the contents of the datasets. For classification tasks, the combination of scikit-learn (Pedregosa et al., 2011) and Pandas (Wes McKinney, 2010) can provide preprocessing, data exploration, powerful modeling, and error analysis. However, chunking tasks like named entity recognition (NER) pose a challenge for data workflows and error analysis. While NER is often treated like a sequence-labeling problem like part of speech (POS) tagging, unlike POS tagging, the annotation and evaluation are performed at the chunk level, not individual tokens.

For example, if a sentence begins "Alan Turing was...", an NER task may require that *Alan Turing* is correctly identified as a mention of a person's name. Less or no credit would be received for identifying just the person name *Alan* or separately identifying the names *Alan* and *Turing* without noting that they form a single unit. Typically, each mention (also called an entity, phrase, or chunk) is encoded using BIO encoding, so for example *Alan Turing* might receive the tags `B-PER I-PER` to reflect the beginning and continuation of a mention of type person, while non-mention tokens are tagged `O`.[1]

The nature of chunking tasks means that every step of data processing is necessarily more complicated than traditional classification tasks. Unlike a per-token classification task, looking at the individual token labels is a poor summary of the dataset. Scoring is more difficult as the scorer must interpret a sequence of tagged tokens as mentions, which becomes non-trivial when the tags produced by a system do not follow the norms of the mention encoding method (e.g., BIO) used (Lignos and Kamyab, 2020).

This paper describes the SeqScore toolkit and its applications for validating, summarizing, and transforming NER data. A previous publication (Palen-Michel et al., 2021), introduced SeqScore and described its value as a reproducibility-focused NER scorer. While this paper is also about SeqScore, it has a different focus. We describe the development of new features for SeqScore and what was needed to extend it from being just a scorer to a more complete toolkit for working with NER data. We discuss new feature development on SeqScore and the process of expanding it to fill gaps common in NER workflows. In addition to discussing the details of SeqScore, we discuss the challenges of trying to build open-source software for research.

2 Why a Toolkit?

For decades, NER researchers have been able to be productive without any popular toolkits for working with NER data. There has been a commonly-used scorer, conlleval, which was provided for the 2002–3 CoNLL shared tasks (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003), available for two decades now. While standard-

[†]Corresponding author.
[*]Equal contribution.

[1]Lester (2020) provides a more detailed explanation of common encodings and their intricacies in the context of software development.

izing around a single scorer provides significant benefits (Lignos and Kamyab, 2020; Post, 2018), conlleval is not actively maintained and has not been updated as approaches to NER have changed.

While common scorers and shared tasks are uniquely capable of uniting the research community, there is still a vital need for tools for tools for many stages of working with NER data. For example, given a dataset, how can we examine it? How can we determine what mention encoding it uses and whether it was used consistently? How can we modify it efficiently? How can we examine performance beyond just a few numbers? SeqScore aims to provide efficient, command-line solutions to these problems.

The most similar software package to SeqScore is iobes (Lester, 2020). While the two projects began development concurrently, iobes was released first and has previously been published at NLP-OSS. The iobes package is designed for API-level access and manipulation of spans. SeqScore is focused on a command-line interface for scoring NER data and performing common manipulations on that data. Both provide logic around converting chunk encodings (BIO, BIOES, etc.) to and from mentions.

3 SeqScore's Features

This section describes the features of SeqScore, focusing on the newest features that enable it to assist in many NER data workflows. Previous work (Palen-Michel et al., 2021) has described the scoring features of SeqScore, so they are not discussed in detail in this paper. SeqScore is released via PyPI (`https://pypi.org/project/seqscore/`) and development occurs on GitHub (`https://github.com/bltlab/seqscore`).

3.1 Overview

SeqScore is accessed via a command-line interface (CLI), and like git provides a command for each action. After running `pip install seqscore`, the `seqscore` script is now available. Table 1 lists SeqScore's commands and their purposes.

All SeqScore commands share a common set of capabilities. The most important is robust reading and writing of CoNLL-style NER formats. SeqScore supports several options to work with a wide variety of data files: setting the file encoding (older files often use ISO-8859-1), ignoring comment lines (which some files use for sentence provenance information), and automatic detection of field delimiters (older files use space, newer ones use tabs). Different strategies can be set regarding how to deal with invalid label transitions like `O I-PER` in BIO (for more details see Palen-Michel et al., 2021). SeqScore can maintain or discard the document boundaries specified using `-DOCSTART-` sentences inside CoNLL-format files, which enables scoring a reference with document boundaries against system output without them.

While each of these features is simple, those attempting to write "quick scripts" to manipulate NER data often find them to be stumbling blocks. For example, many older workflows used tools like `cut` and `paste` to extract and replace NER labels; these encounter problems when dealing with comment lines, inconsistent delimiters, document boundaries in one file but not the other, etc.

3.2 Validation

One of the most common questions that arises when dealing with NER data is determining which mention encoding (BIO, etc.) is used and whether it has been used consistently. While BIO is currently the most commonly-used encoding for dataset creation, many papers describing datasets do not explicitly state what the encoding is. Many older datasets use IOB, often erratically.[2]

An example of running the validate command on a file `train.bio` would be: `seqscore validate --labels BIO train.bio`. The mention encoding in use must be explicitly given; it is important that users be sure of the mention encoding they are using.

As part of an effort to test SeqScore to make sure it can reliably load a variety of datasets, we collected a mix of recent and older datasets and validated them. The following datasets passed validation without any modifications: NYTK-NerKor+Cars-OntoNotes++ (Novák and Novák, 2022; Simon and Vadász, 2021), TurkuNLP (Luoma et al., 2020), GermanLER (Leitner et al., 2019; Leitner, 2019), TweebankNER (Jiang et al., 2022), HiNER (Murthy et al., 2022), GermEval 2014

[2]Our experiments in validating older datasets led to interesting findings. There seems to be disagreement on how IOB encoding should be implemented. Some data files use `B-` only when strictly necessary, that is when two adjacent mentions of the same entity type appear, as would be for the fragment "[Australian]MISC [Davis Cup]MISC captain" from the CoNLL-02 English data. Others use `B-` for the second mention of two adjacent mentions, even if the entity types are different. SeqScore currently only allows the former variety to pass IOB validation, but this may change in future releases.

Command	Purpose
`convert`	Convert between different mention encodings (BIO, BIOES, etc.)
`count`	Show counts of the mentions in a file in descending order
`process`	Modify the mentions in a file by choosing which entity types to keep/remove or mapping entity types
`repair`	Correct invalid label transitions
`score`	Produce a score or a summary of the scoring events (false positives, etc.)
`summarize`	Give a high-level summary of a dataset that includes its length and the count of each entity type
`validate`	Check whether a file contains any invalid labels or invalid label transitions

Table 1: Description of SeqScore commands

(Benikova et al., 2014), CoNLL-03 English and German (Tjong Kim Sang and De Meulder, 2003, we used the 2006 German data re-release), CoNLL-02 Dutch (Tjong Kim Sang, 2002), and Europarl annotation (Agerri et al., 2018).

Three datasets contained invalid label transitions, but using SeqScore's `repair` command could be converted to valid versions that pass validation: CoNLL-02 Spanish (Tjong Kim Sang, 2002), KazNERD (Yeshpanov et al., 2022), and MultiCoNER II (Fetahu et al., 2023).

SeqScore's validation tool helped identify more significant issues with other datasets. When validating the BIO-encoded MasakhaNER 1.0 (Adelani et al., 2021) dataset, we found sentences beginning with `I-` labels that appeared to be a continuation of a mention at the end of the previous sentence. When we investigated, we discovered that after creation of the original dataset, long sentences were split without regard to mention boundaries in order to meet the maximum sentence length requirements of the models used in their study. We were able to locate an earlier version of the data in their GitHub repository that did not have these additional sentence breaks, and that data passed validation.

Other datasets passed validation after modifications were made to them. MahaNER (Litake et al., 2022) has tags of the form `BPER` instead of `B-PER`. After changing the tags to add `-`, SeqScore's `repair` command was used to correct invalid label transitions. The tags in KIND (Paccosi and Palmero Aprosio, 2022) are "bare" tags like `PER`. After changing all tags to IO tags like `I-PER`, the dataset was successfully validated as IO.

The only dataset we found to be unusable was SiNER (Ali et al., 2020), as it appears to have some text processing issue resulting in some of the lines having tokens but no label, and others the reverse.

3.3 Data Modification

While core entity types likes person, organization, and location appear in many NER datasets, different datasets use different ontologies. Often some entity types are annotated less reliably, like MISC in CoNLL 2002–3, and others may simply be of less interest, like DATE in MasakhaNER.

SeqScore supports specifying either a set of entity types to keep or remove. For example, to include only PER, LOC, and ORG, the user can run `seqscore process --keep-types PER,LOC,ORG input.bio output.bio`. Similarly, to remove the DATE type, the user can run `seqscore process --remove-types DATE input.bio output.bio`.

Another common task is collapsing a fine-grained set of types into a smaller set. For example, the MultiCoNER II dataset (Fetahu et al., 2023) is annotated with 33 fine-grained types which can be mapped to 6 coarse-grained types. The annotation is provided using fine-grained types, so to evaluate for the coarse types, the types must be mapped.

SeqScore supports this type mapping using a JSON file. For example, this JSON can be used to convert to the higher-level types PROD and LOC from fine-grained types: `{"PROD": ["Clothing", "Drink", "Food", "OtherPROD", "Vehicle"], "LOC": ["Facility", "HumanSettlement", "OtherLOC", "Station"]}`. This mapping can be used as follows: `seqscore process --type-map map.json input.bio output.bio`.

3.4 Error Analysis at a Glance

In text classification tasks, confusion matrices allow for quick understanding of error patterns in a system's output. For a chunking task like NER, it is difficult to define exactly what a confusion matrix should look like. For SeqScore, we attempted to come up with a way to summarize the errors in a system's output in a way similar to identifying the "hot spots" in a heat map of the confusion matrix. When scoring, SeqScore can produce a table of false positives and negatives sorted by descending frequency by using the `--error-counts` flag: `seqscore score --error-counts --labels BIO --reference ref.bio pred.bio`.

Count	Error	Type	Text
7	FP	MISC	ALPINE
5	FN	ORG	Real Madrid
5	FN	ORG	Barcelona
4	FP	LOC	Tasmania
4	FP	LOC	Victoria
4	FP	MISC	National Hockey
4	FP	MISC	League
4	FP	MISC	ATLANTIC DIVISION
4	FP	MISC	PACIFIC DIVISION
4	FP	LOC	Santa Fe
4	FN	ORG	Victoria
4	FN	ORG	Tasmania
4	FN	ORG	National Hockey
4	FN	ORG	League
4	FN	MISC	ATLANTIC
4	FN	LOC	PACIFIC
4	FN	ORG	Santa Fe
3	FP	MISC	World Cup
3	FP	ORG	William Hill
3	FP	MISC	Italian
3	FP	MISC	EST
3	FP	MISC	Conservative
3	FP	MISC	SKIING-WOMEN
3	FN	MISC	SKIING-WORLD CUP
3	FN	ORG	NFL

Table 2: Most-frequent false positive (FP) and false negative (FN) errors identified using `seqscore score --error-counts`

Table 2 shows output of this command from an NER model based on XLM-R (Conneau et al., 2020) and fine-tuned on CoNLL++ English data using FLERT (Schweter and Akbik, 2020). It immediately shows that some of the most-repeated errors happen in all-caps contexts. The output also suggests that sports is a problem domain, with leagues, sub-leagues, and clubs appearing in both false positives and negatives. Looking at this output allowed us to identify problems with the annotation of *National Hockey League* in the CoNLL++ test data; a deeper look revealed that improper sentence boundaries in the gold data repeatedly resulted in split mentions of *National Hockey* and *League*.

As most papers reporting NER scores do not report any error analysis, we hope the ease with which the most frequent errors can be looked at in SeqScore will help researchers at least identify the largest sources of error.

4 Design Challenges

Explicit or Implicit? Unlike conlleval, which will score many mention encodings without any direction from the user—even encodings it does not support (Akbik et al., 2019, footnote 2)—SeqScore requires users to be specify the mention encoding and how they want invalid transitions to be repaired. This can be confusing to new users, because if they do not specify a repair method, scoring may raise an error. SeqScore has always erred on the side of making users be explicit and avoiding any silent defaults that could affect the results, but this comes at a price of some user frustration.

Limiting Scope While SeqScore is so far the most richly-featured toolkit for working with NER files, we have intentionally limited the scope of its capabilities where there is risk of misuse. One example is scoring NER in cases where the reference and system output may disagree in the tokens of a sentence, such as when performing NER on the output of speech recognition. SeqScore currently insists that the reference and system output have the same text to avoid issues where sentences have become misaligned between the two, and this cannot be disabled. Providing a flag to disable this check could result in users specifying it "just in case" to make sure scoring never raises an error, potentially leading to incorrect scores. Benaicha et al. (2023) forked SeqScore for their study of NER on speech so they could score more flexibly.

Test Coverage While SeqScore stands apart from much research software in having a test suite and code coverage instrumentation, as the complexity of the toolkit has increased, so has the time required for tests to keep up with functionality. While very time-intensive to maintain, writing tests that exercise the command-line interface has been essential to avoiding regressions. SeqScore uses click[3] to implement the CLI, and testing is greatly aided by its `CliRunner` class which allows direct invocation of the CLI in unit tests.

SeqScore's test coverage stands at 95%, but it will take substantial effort to reach 100%. A handful of warnings and error cases are not exercised by the current tests due to the high time cost of developing inputs that would reach them and maintaining these inputs as the codebase changes.

Performance SeqScore is written highly defensively to protect against user errors that could result in incorrect evaluation results. This unfortunately comes at the cost of speed; SeqScore is slower at scoring and processing long files than other scorers. While we are interested in improving speed by using profiling, we are unwilling to optimize for speed at the expense of safety.

[3] https://palletsprojects.com/p/click/

5 Conclusion

SeqScore provides a feature-rich toolkit for working with NER data, and we believe it will enable easier and more reproducible NER research. As more users adopt SeqScore, we look forward to addressing their needs and the bugs they find.

Development of SeqScore is ongoing. A major area of interest is developing a stable API for scoring. Unlike iobes and seqeval (Nakayama, 2018), the primary use case of SeqScore has been through the command line. We plan to add a stable API before a version 1.0 release.

References

David Ifeoluwa Adelani, Jade Abbott, Graham Neubig, Daniel D'souza, Julia Kreutzer, Constantine Lignos, Chester Palen-Michel, Happy Buzaaba, Shruti Rijhwani, Sebastian Ruder, Stephen Mayhew, Israel Abebe Azime, Shamsuddeen H. Muhammad, Chris Chinenye Emezue, Joyce Nakatumba-Nabende, Perez Ogayo, Aremu Anuoluwapo, Catherine Gitau, Derguene Mbaye, Jesujoba Alabi, Seid Muhie Yimam, Tajuddeen Rabiu Gwadabe, Ignatius Ezeani, Rubungo Andre Niyongabo, Jonathan Mukiibi, Verrah Otiende, Iroro Orife, Davis David, Samba Ngom, Tosin Adewumi, Paul Rayson, Mofetoluwa Adeyemi, Gerald Muriuki, Emmanuel Anebi, Chiamaka Chukwuneke, Nkiruka Odu, Eric Peter Wairagala, Samuel Oyerinde, Clemencia Siro, Tobius Saul Bateesa, Temilola Oloyede, Yvonne Wambui, Victor Akinode, Deborah Nabagereka, Maurice Katusiime, Ayodele Awokoya, Mouhamadane MBOUP, Dibora Gebreyohannes, Henok Tilaye, Kelechi Nwaike, Degaga Wolde, Abdoulaye Faye, Blessing Sibanda, Orevaoghene Ahia, Bonaventure F. P. Dossou, Kelechi Ogueji, Thierno Ibrahima DIOP, Abdoulaye Diallo, Adewale Akinfaderin, Tendai Marengereke, and Salomey Osei. 2021. MasakhaNER: Named entity recognition for African languages. *Transactions of the Association for Computational Linguistics*, 9:1116–1131.

Rodrigo Agerri, Yiling Chung, Itziar Aldabe, Nora Aranberri, Gorka Labaka, and German Rigau. 2018. Building named entity recognition taggers via parallel corpora. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Alan Akbik, Tanja Bergmann, and Roland Vollgraf. 2019. Pooled contextualized embeddings for named entity recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 724–728, Minneapolis, Minnesota. Association for Computational Linguistics.

Wazir Ali, Junyu Lu, and Zenglin Xu. 2020. SiNER: A large dataset for Sindhi named entity recognition. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2953–2961, Marseille, France. European Language Resources Association.

Moncef Benaicha, David Thulke, and M. A. Tuğtekin Turan. 2023. Exploring spoken named entity recognition: A cross-lingual perspective. ArXiv preprint 2307.01310.

Darina Benikova, Chris Biemann, Max Kisselew, and Sebastian Pado. 2014. GermEval 2014 named entity recognition shared task: companion paper. In *Workshop Proceedings of the 12th edition of the KONVENS conference*, pages 104–112.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Besnik Fetahu, Sudipta Kar, Zhiyu Chen, Oleg Rokhlenko, and Shervin Malmasi. 2023. SemEval-2023 task 2: Fine-grained multilingual named entity recognition (MultiCoNER 2). In *Proceedings of the The 17th International Workshop on Semantic Evaluation (SemEval-2023)*, pages 2247–2265, Toronto, Canada. Association for Computational Linguistics.

Hang Jiang, Yining Hua, Doug Beeferman, and Deb Roy. 2022. Annotating the Tweebank corpus on named entity recognition and building NLP models for social media analysis. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 7199–7208, Marseille, France. European Language Resources Association.

Elena Leitner. 2019. Eigennamen- und Zitaterkennung in Rechtstexten. Master's thesis, Universität Potsdam, Potsdam.

Elena Leitner, Georg Rehm, and Julian Moreno-Schneider. 2019. Fine-grained Named Entity Recognition in Legal Documents. In *Semantic Systems. The Power of AI and Knowledge Graphs. Proceedings of the 15th International Conference (SEMANTiCS 2019)*, number 11702 in Lecture Notes in Computer Science, pages 272–287, Karlsruhe, Germany. Springer. 10/11 September 2019.

Brian Lester. 2020. iobes: Library for span level processing. In *Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS)*, pages 115–119, Online. Association for Computational Linguistics.

Constantine Lignos and Marjan Kamyab. 2020. If you build your own NER scorer, non-replicable results will come. In *Proceedings of the First Workshop on Insights from Negative Results in NLP*, pages 94–99, Online. Association for Computational Linguistics.

Onkar Litake, Maithili Ravindra Sabane, Parth Sachin Patil, Aparna Abhijeet Ranade, and Raviraj Joshi. 2022. L3Cube-MahaNER: A Marathi named entity recognition dataset and BERT models. In *Proceedings of the WILDRE-6 Workshop within the 13th Language Resources and Evaluation Conference*, pages 29–34, Marseille, France. European Language Resources Association.

Jouni Luoma, Miika Oinonen, Maria Pyykönen, Veronika Laippala, and Sampo Pyysalo. 2020. A broad-coverage corpus for Finnish named entity recognition. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4615–4624, Marseille, France. European Language Resources Association.

Rudra Murthy, Pallab Bhattacharjee, Rahul Sharnagat, Jyotsana Khatri, Diptesh Kanojia, and Pushpak Bhattacharyya. 2022. HiNER: A large Hindi named entity recognition dataset. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4467–4476, Marseille, France. European Language Resources Association.

Hiroki Nakayama. 2018. seqeval: A Python framework for sequence labeling evaluation. Software available from `https://github.com/chakki-works/seqeval`.

Attila Novák and Borbála Novák. 2022. NerKor+Cars-OntoNotes++. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 1907–1916, Marseille, France. European Language Resources Association.

Teresa Paccosi and Alessio Palmero Aprosio. 2022. KIND: an Italian multi-domain dataset for named entity recognition. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 501–507, Marseille, France. European Language Resources Association.

Chester Palen-Michel, Nolan Holley, and Constantine Lignos. 2021. SeqScore: Addressing barriers to reproducible named entity recognition evaluation. In *Proceedings of the 2nd Workshop on Evaluation and Comparison of NLP Systems*, pages 40–50, Punta Cana, Dominican Republic. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Stefan Schweter and Alan Akbik. 2020. FLERT: Document-level features for named entity recognition.

Eszter Simon and Noémi Vadász. 2021. Introducing NYTK-NerKor, a gold standard Hungarian named entity annotated corpus. In *Text, Speech, and Dialogue: 24th International Conference, TSD 2021, Olomouc, Czech Republic, September 6–9, 2021, Proceedings 24*, pages 222–234. Springer.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Wes McKinney. 2010. Data Structures for Statistical Computing in Python. In *Proceedings of the 9th Python in Science Conference*, pages 56 – 61.

Rustem Yeshpanov, Yerbolat Khassanov, and Huseyin Atakan Varol. 2022. KazNERD: Kazakh named entity recognition dataset. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 417–426, Marseille, France. European Language Resources Association.