If You Build Your Own NER Scorer, Non-replicable Results Will Come

Constantine Lignos **Marjan Kamyab**
Brandeis University
415 South St.
Waltham, MA, 02453, USA
{lignos,marjankamyab}@brandeis.edu

Abstract

We attempt to replicate a named entity recognition (NER) model implemented in a popular toolkit and discover that a critical barrier to doing so is the inconsistent evaluation of improper label sequences. We define these sequences and examine how two scorers differ in their handling of them, finding that one approach produces F1 scores approximately 0.5 points higher on the CoNLL 2003 English development and test sets. We propose best practices to increase the replicability of NER evaluations by increasing transparency regarding the handling of improper label sequences.

1 Introduction

The goal of this paper is to demonstrate an issue that complicates the comparison and replication of named entity recognition (NER) systems. Standard F1-based evaluation of NER models in the manner made popular by the CoNLL 2002–3 shared tasks (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003) requires decoding a sequence of per-token labels into entity mentions and computing precision and recall by evaluating the types and spans of the mentions. While there are popular scorer implementations and popular NER toolkits, there is little transparency regarding the *exact* process that scorers use to decode label sequences.

In the case where all label sequences are properly formed, this lack of transparency should have no impact; all correct decoding processes should produce the same result. However, many systems can produce what we call *improper* label sequences (see Section 3.2), and different approaches to decoding improper label sequences for evaluation produce different scores. As the processes for label decoding have not been standardized and are often undocumented, comparisons between reported scores may not be fair, and replication of those scores can prove difficult.

2 Our negative result

We did not intend to begin a project on NER evaluation reproducibility. The discovery process for our negative result began with three separate research projects encountering the same issue: higher-than-expected F1 scores for a certain class of models implemented in a popular NER toolkit, NCRF++ (Yang et al., 2018; Yang and Zhang, 2018).

One of the authors of this paper attempted to reimplement NCRF++'s models as a learning exercise. The reimplementation yielded similar F1 scores to NCRF++ when a CRF output layer was used but produced lower F1 scores when using softmax output. After the other author found the same result in an independent reimplementation, we turned our attention to the only commonality between our implementations: an open-source external scorer. We consulted with researchers in our lab using NCRF++ for two other projects, and a consistent story began to emerge: when scoring softmax models, NCRF++'s internal scorer produced higher scores than other NER scorers.

Our negative result was a failure to replicate the performance of NCRF++, and the cause of this failure was that we did not understand its approach to evaluation. NCRF++'s scorer differs from others in how it treats improper label sequences, and its approach consistently produces higher F1 scores for softmax-output models. The handling of these sequences is effectively undefined behavior for an NER scorer; there is no single correct strategy. In this paper, we quantify the impact of those strategies and propose an evaluation approach that would improve reproducibility by requiring explicit, transparent handling of improper label sequences.

Supplemental material and the resources needed to replicate this study are available at `https://lignos.org/repro-ner`.

Proceedings of the First Workshop on Insights from Negative Results in NLP, pages 94–99
Online, November 19, 2020. ©2020 Association for Computational Linguistics
ISBN 978-1-952148-66-8

3 Entity encoding and decoding

3.1 Proper label sequences

As the focus of this venue is on insights from negative results and not the finer points of NER systems, we will first review the process of entity encoding and decoding. Consider the following sentence fragment from the CoNLL 2003 English NER data:

> [Australian]MISC [Davis Cup]MISC captain [John Newcombe]PER.

In this fragment, three entity mentions are annotated. Table 1 shows three well-known approaches to encoding these tokens as a label sequence.

Encoding	Labels					
IOB	I-MISC	B-MISC	I-MISC	O	I-PER	I-PER
BIO	B-MISC	B-MISC	I-MISC	O	B-PER	I-PER
BIOES	S-MISC	B-MISC	E-MISC	O	B-PER	E-PER

Table 1: Proper entity encodings for the tokens of the string *Australian Davis Cup captain John Newcombe*.

Note that in BIO (Begin, Inside, Outside) encoding, every mention begins with a `B` label; in IOB encoding, mentions begin with `I` except where necessary to differentiate from the continuation of a preceding same-type mention by using a `B` label (e.g., `I-PER B-PER` for two adjacent single-token names). In the BIOES encoding, single-token entities use a single `S` label, and multi-token entities begin with `B`, end with `E`, and use `I` for everything but the first and last tokens.[1]

3.2 Improper label sequences

This paper is concerned with the implications for evaluation in NER when unexpected label sequences are produced. While there does not appear to be any standard term for this phenomenon, we define an *improper* label sequence as one where the label sequence does not conform to a sequence of labels allowed by the encoding. Consider the improper label sequences given in Table 2.

In all three cases, it is possible to infer the likely "intent" of the system that produced these label sequences. For IOB and BIO encodings, the `conlleval` scorer (Tjong Kim Sang, 2004) used for the CoNLL 2002–3 NER evaluations (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003) would effectively "repair"[2] these label sequences, such that they would produce the same entity mentions as the proper encodings in Table 1. We refer to this approach to interpreting improper label sequences as *CoNLL-style* in this paper.

Encoding	Labels					
IOB	I-MISC	B-MISC	I-MISC	O	**B-PER**	I-PER
BIO	B-MISC	B-MISC	I-MISC	O	**I-PER**	I-PER
BIOES	S-MISC	B-MISC	**I-MISC**	O	B-PER	E-PER

Table 2: Improper entity encodings for the tokens of the string *Australian Davis Cup captain John Newcombe*, with improper labels identified using bold.

An alternative would be to interpret the label sequence more strictly. An obvious approach for BIO encoding would be to *only* begin a mention when a `B` label was encountered. In this example, only two mentions would be created, as the final two tokens do not have a proper beginning label.

The `conlleval` scorer does not support the BIOES/IOBES encodings, but `seqeval` (chakki, 2019), a Python library that replicates `conlleval`'s BIO decoding, supports them and uses similar CoNLL-style repair logic for improper BIOES/IOBES sequences. We return to the issue of BIOES/IOBES decoding in Section 5.1.

3.3 Softmax output

Since at least HMM-based NER systems of the late 90s, sequence NER models have identified the most likely label sequence by taking into account the relationship between a label and preceding labels. The most common modern approach to modeling this relationship is to use a conditional random field (CRF), which can learn to avoid producing improper label sequences or be forced to, either through manual manipulation of its weights or forbidding improper transitions entirely.

With the advent of neural models capable of capturing substantial contextual information before the output layer, it is now feasible to create relatively good NER models without a CRF by using softmax to select the highest-scoring label for each

[1]Historically, it has not been possible to identify *with certainty* the entity encoding an NER study uses based on the acronym used. In early work there was confusion between the IOB (IOB1, Ramshaw and Marcus, 1995; Tjong Kim Sang and Veenstra, 1999) and BIO (IOB2) encodings, and later similar confusion between BIOES and IOBES. Readers may disagree regarding precisely what entity encodings are signified by these acronyms. We do not discuss BMES and BILOU in this paper and consider them isomorphic to BIOES.

[2]It is not clear whether it was an explicit design goal to repair these sequences or they are simply the consequence of implementing a universal decoder for IOB and BIO encodings. Readers interested in examining the decoding logic should inspect the `startOfChunk` and `endOfChunk` functions.

token independently, which is significantly faster. While the model will still indirectly learn to prefer proper label sequences, there is no explicit representation of the sequential relationship between labels. When sufficiently trained, a model with CRF output rarely produces improper label sequences, but softmax-output models do so more frequently.

4 Results

The goal of our experiments is to estimate the increase in F1 scores that can be attributed to use of NCRF++'s internal decoding compared to CoNLL-style decoding when evaluating softmax-output models. We trained models using NCRF++ after modifying it to additionally use an external CoNLL-style scorer, `seqeval`.[3] We report scores from the internal and external scorer. We use a bidirectional LSTM architecture at the word level and test multiple character-level architectures: a bi-LSTM, CNN, and no character-level representation. We use the same hyperparameters, pretrained word embedding (GLoVe 100d), and data (CoNLL 2003 English) used by Yang et al. (2018). Each configuration was run ten times using different random initializations (seeds 0–9). The test set was evaluated using the model from the epoch that attained the highest development set F1 (as scored internally by NCRF++) during training.

Table 3 gives F1 scores for entity mentions for BIOES and BIO encodings across all character-level architectures, including values previously reported by Yang et al. (2018). When using NCRF++'s internal scorer, our results are close to the those previously reported. However, evaluating the same output with an external scorer leads to lower scores.

Table 4 reports the distribution of the increase in F1 scores (NCRF++'s internal score minus the external score) for each system output across encodings and evaluation sets. We computed how much NCRF++'s scoring procedure increases F1 for each run's output and then average across all runs (character-level architectures and random initializations). To demonstrate the statistical reliability of this increase, we performed a Wilcoxon signed-rank test—a non-parametric version of the paired t-test—for each combination of encoding and evaluation set. All four p-values were below 0.0001, and the 95% confidence intervals of the differences were .49–.58 (BIOES) and .38–.43 (BIO) for development, .54–63 (BIOES) and .48–.56 (BIO) for test. In summary, for softmax models, the NCRF++ internal scorer produces scores approximately half a point of F1 higher.

All scores reported so far have been from converged models. It is also of interest to explore what the increase in scores looks like during training. Figure 1 gives the increase in development set F1 scores across all training epochs for all configurations we ran, displaying 3,000 points per encoding. Early in training, NCRF++'s internal scorer can produce F1 scores several points higher than a CoNLL-style scorer, presumably due to producing a high number of improper label sequences pre-convergence. Crucially, in all 6,000 development set epochs we evaluated, NCRF++'s internal scorer *always* reported a higher F1 than `seqeval` in our evaluation of softmax-output models.

5 Discussion

5.1 Analysis

Before we discuss the insights gained from our study, it is important contextualize our findings. Our goals are to motivate the establishment of standard evaluation practices for NER, explain the impact of improper label sequence decoding, and encourage authors to be transparent about their approach to evaluation.

We must emphasize that we are not claiming that the way that NCRF++'s scorer decodes improper label sequences is incorrect; it is one of many possible ways of doing so, and we do not wish to single out NCRF++ or its authors specifically for criticism. NCRF++'s approach to decoding improper label sequences is, however, different than the popular approach defined by the widely-used `conlleval` scorer for the CoNLL 2002–3 NER shared tasks (later faithfully reimplemented and extended by `seqeval`), and thus one must exercise caution when comparing scores it generates to those of other scorers.

While we find the NCRF++ entity mention decoder difficult to fully understand, our inspection of the code leads us to believe that it will only begin a mention when the proper tag—`B` for BIO, `B` or `S` for BIOES—is supplied. It is effectively removing some improper label sequences by treating some labels as if they were `O`.

[3]We selected `seqeval` after reviewing its label decoding procedure and confirming that produces the same scores for BIO as `conlleval`. We chose it because it supports both BIO and BIOES encoding and provides greater numerical precision than `conlleval`.

			BIOES			BIO		
Source	*N*	Scorer	No Char.	Char. LSTM	Char. CNN	No Char.	Char. LSTM	Char. CNN
Reported	5	NCRF++	$88.49 \pm .17$	$90.77 \pm .06$	$90.60 \pm .11$	-	-	-
Reproduction	10	NCRF++	$88.78 \pm .28$	$90.76 \pm .12$	$90.62 \pm .15$	$88.41 \pm .21$	$90.49 \pm .15$	$90.46 \pm .58$
Reproduction	10	External	$88.21 \pm .27$	$90.20 \pm .09$	$89.99 \pm .19$	$87.98 \pm .21$	$89.93 \pm .19$	$89.88 \pm .21$

Table 3: Means and standard deviation of test set F1 for each tested configuration and from previously reported results (Yang et al., 2018, Table 4). Empty cells indicate configurations without previously reported scores. *N* gives the number of runs used to compute each value in the row (e.g., each mean was computed over *N* values).

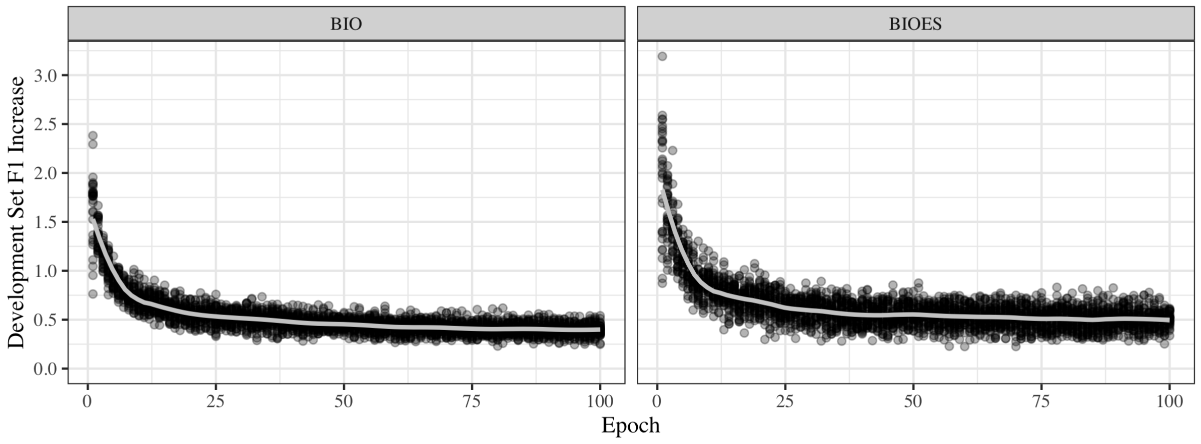

Figure 1: Increase in development set F1 due to NCRF++'s internal scorer for softmax output models across all training epochs, with a local regression (LOESS) fit line.

Data Set	*N*	BIOES	BIO
Development	30	0.53 ± 0.11	0.40 ± 0.06
Test	30	0.58 ± 0.12	0.52 ± 0.11

Table 4: Means and standard deviations for softmax models of the increase in F1 due to using NCRF++'s internal scorer.

Why would it be advantageous for a system to decode in this way? Unlike in Table 2, where repairing the improper sequences would result in correct answers, many improper sequences represent generalization errors. For example, consider a system using a BIO encoding that observed *Pat Jones* as `B-PER I-PER` frequently in training and must predict labels for *Charity Jones*, annotated as a person in the gold standard. If it has never seen the token *Charity* with the label `B-PER` and has seen *Jones* frequently with the label `I-PER`, it might predict the tag sequence `O I-PER`. A lenient CoNLL-style decoder would decode this as a mention of type `PER` for the token *Jones*, which would result in a false alarm (precision error) and a miss (recall error).

However, a stricter BIO decoder like NCRF++'s would predict no mentions, resulting in a miss but no false alarm. For softmax output, it appears to be universally advantageous to decode this way; for the 6,000 epochs we scored using the development set, NCRF++'s scorer always gave a higher F1.

Our examples have focused on decoding BIO sequences, partly because `conlleval` set a standard approach to doing so almost two decades ago. Repairing improper BIOES label sequences is much more complex; for example, in the sequences `B-PER E-PER E-PER` and `S-PER I-PER E-PER`, how many entity mentions should be created? There is not a single answer, but `seqeval` implements an approach in the spirit of what `conlleval` does for BIO, and NCRF++ appears to implement a stricter decoder.

While we were not able to find any published or unpublished papers specifically discussing the effect of decoding improper label sequences on F1, as we prepared this paper for submission we discovered a closed GitHub issue[4] opened by Mike Kroutikov pointing out that the way NCRF++ decodes label sequences does not match other NER

[4] `https://github.com/jiesutd/NCRFpp/issues/87`

systems. Reviewing this issue and other issues opened against NCRF++, it is clear that we are not the first to identify its departure from CoNLL-style label decoding. In addition to opening an issue, Kroutikov (2019) blogged about the many potential ways to decode BIOES and examined the approaches taken by other NER implementations.

Our findings raise the question of whether NCRF++'s internal scorer also differs from the external scorer when evaluating models that use CRF output. We performed a post-hoc study where we repeated the same set of experiments we performed for softmax-output models with ones that used a CRF. While NCRF++'s scorer does not give the exact same scores as the external scorer, the increase in F1 attributable to NCRF++'s scoring procedure is quite small when measured on the development set (BIOES 0.0032 ± 0.0040; BIO 0.0072 ± 0.0073), and test set (BIOES -0.015 ± 0.020; BIO 0.0040 ± 0.0051). The experiments using BIOES encoding and evaluating on the test set are the only ones we performed in which on average NCRF++'s internal scorer produced lower scores than the external one.

5.2 Insights

What insights have we gained from our negative result? First, we have clarified how well softmax-output NER models perform when evaluated using CoNLL-style label decoding, which gives lower scores than NCRF++'s internal scorer. The issue of improper label decoding is unlikely to significantly affect any state of the art results, which do not generally use softmax output, but may affect decision-making for NER system designers exploring whether adding a CRF to a system is worth the performance penalty.

Second, we believe our study will help users of a popular toolkit—which received the COLING 2018 "Most reproducible" best paper award—understand how it computes scores and why attempts at replication will fail if they do not also replicate NCRF++'s approach to improper label sequence decoding.

More broadly, the insights from this negative result highlight for all NLP researchers the importance of using a standard evaluation procedure. When possible, using widely-used, well-documented scorers enables fair comparisons of scores across systems.

6 A vision for NER evaluation

Our study leads us to propose a vision for NER evaluation as follows:

1. We should have a well-tested, well-documented, open-source scorer which has been developed independently of any particular model. This scorer should only accept properly-formed label sequences, avoiding the question of the "right" way to decode improper label sequences.
2. The scorer should be accompanied by implementations of standard processes for converting improper label sequences into proper ones. These approaches should include the CoNLL-style approach and a more strict one, like NCRF++'s.
3. Work which wants to use an alternative approach to converting improper label sequences to proper ones should contain a documented, replicable process for doing so.

This procedure separates the process of scoring from the process of interpreting improper label sequences. It also suggests a new research avenue of designing methods for optimally converting improper label sequences into proper ones.

Until the tools required for our vision—perhaps something like SacreBLEU (Post, 2018)—are developed, we recommend `seqeval` as the best solution for NER scoring. It is an easy-to-inspect, faithful reimplementation of `conlleval`. Unlike `conlleval`, it can be called directly from Python and does not truncate scores, avoiding systematic downward bias when aggregating them.

Regarding extensions to this study, while we evaluated on the CoNLL 2003 English data to compare with Yang et al. (2018), evaluating in more languages is essential. Looking beyond the other CoNLL 2002–3 languages, testing against smaller annotated data sets and in lower-resourced languages may reveal more complexity to the problem of decoding improper sequences.

Acknowledgments

Thanks to Elena Álvarez Mellado and Jingxuan Tu, whose usage of NCRF++ for other projects helped us identify the negative result featured here. Thank you to Chester Palen-Michel and three anonymous reviewers for providing feedback on this paper.

References

chakki. 2019. seqeval. Version 0.0.12, https://github.com/chakki-works/seqeval.

Mike Kroutikov. 2019. 7776 ways to compute F1 for an NER task. http://blog.innodatalabs.com/7776_ways_to_compute_f1_for_ner_task/.

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.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

Erik Tjong Kim Sang. 2004. conlleval. Version 2004-01-26, https://www.clips.uantwerpen.be/conll2002/ner/bin/conlleval.txt.

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.

Erik F. Tjong Kim Sang and Jorn Veenstra. 1999. Representing text chunks. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 173–179, Bergen, Norway. Association for Computational Linguistics.

Jie Yang, Shuailong Liang, and Yue Zhang. 2018. Design challenges and misconceptions in neural sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3879–3889, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jie Yang and Yue Zhang. 2018. NCRF++: An open-source neural sequence labeling toolkit. In *Proceedings of ACL 2018, System Demonstrations*, pages 74–79, Melbourne, Australia. Association for Computational Linguistics.