Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 1704–1711
Marseille, 11–16 May 2020

HardEval: Focusing on Challenging Tokens to Assess Robustness of NER

Gabriel Bernier-Colborne, Philippe Langlais
NRC, RALI-DIRO
Montreal, Canada
gabriel.bernier-colborne@nrc.ca, felipe@iro.umontreal.ca

Abstract

To assess the robustness of NER systems, we propose an evaluation method that focuses on subsets of tokens that represent specific sources of errors: unknown words and label shift or ambiguity. These subsets provide a system-agnostic basis for evaluating specific sources of NER errors and assessing room for improvement in terms of robustness. We analyze these subsets of challenging tokens in two widely-used NER benchmarks, then exploit them to evaluate NER systems in both in-domain and out-of-domain settings. Results show that these challenging tokens explain the majority of errors made by modern NER systems, although they represent only a small fraction of test tokens. They also indicate that label shift is harder to deal with than unknown words, and that there is much more room for improvement than the standard NER evaluation procedure would suggest. We hope this work will encourage NLP researchers to adopt rigorous and meaningful evaluation methods, and will help them develop more robust models.

Keywords: named entity recognition, natural language processing, evaluation, robustness

1. Introduction

Named entity recognition (NER) is one of the most common applications of natural language processing (NLP). Given a text, an NER system is tasked with detecting expressions that refer to named entities, e.g. people, locations or organizations, and predicting the entity type of each detected mention. NER is used in various downstream applications, which typically involve information extraction or retrieval. NER is a relatively well-studied problem, and is often used to benchmark NLP systems.

The standard method used to assess NER systems is an automatic, quantitative evaluation based on a human-annotated dataset, whereby the system's output is compared to the human annotations. Most of the time, systems are trained and tested on data from the same domain and distribution, i.e. disjoint subsets of the same dataset, typically a fixed train/dev/test split. The high scores achieved by modern NLP systems in this setting suggest that their performance is close to, or even better than, human performance on this task. Thus, one might conclude that NER is a "solved problem".

However, cross-domain evaluation, whereby systems are trained and tested on data from different domains, paints a less rosy picture, as the change in the data distribution results in poorer performance. This suggests that NER systems lack robustness, and have a limited ability to learn what a named entity actually is in a way that generalizes across domains.

To assess the robustness of NER systems, we propose an evaluation method that focuses on subsets of tokens that represent specific sources of errors: unknown words and label shift or ambiguity. These sources of errors arise more frequently when there is a shift in the data distribution, but are also present in the single-distribution setting.

In this paper, we examine how these two phenomena manifest themselves within two widely-used NER benchmarks, then we apply the proposed evaluation method to obtain an assessment of the robustness of various approaches to NER, conducting both in-domain and out-of-domain evaluations. Results show that modern NER systems still have a limited ability to handle unknown words and label shift, and it appears that label shift is much harder to deal with than unknown words.

We hope this work will encourage NLP researchers to adopt rigorous and meaningful evaluation methods, and to stop relying on those that would suggest that NER is a solved problem. In that spirit, we have made our code publicly available.[1]

2. Data

For this study, we focused on two widely-used benchmark datasets for NER: CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003) and OntoNotes (Hovy et al., 2006; Weischedel et al., 2012).

- **CoNLL** contains newswire texts (including many sports articles), in which four types of named entities are annotated: people (PER), organizations (ORG), locations (LOC), and various other entity types (MISC), which include languages, nationalities, product names, event names, etc.
- **OntoNotes** includes texts from 6 different sources,[2] representing different domains and genres: broadcast conversations (BC), broadcast news (BN), magazines (MZ), newswire (NW), telephone conversation transcripts (TC), and blogs and newsgroups (WB). The entity type classification is more fine-grained than that of CoNLL, comprising 18 different types, which include specific entity types that are subsumed by a generic entity type in CoNLL (e.g. geopolitical entities, which are lumped together with locations in CoNLL), as well as various numeric entities, which are not annotated in CoNLL.

[1] `https://github.com/gbcolborne/ner_eval`.

[2] In the OntoNotes documentation (Weischedel et al., 2012), MZ and NW are lumped together, but we decided to keep them separate, as in (Ghaddar and Langlais, 2018).

Both datasets come pre-tokenized. Labels are encoded using the IOB2 format.[3]
Statistics on the named entity mentions in the training portion of these datasets are shown in Table 1. These include the number of mentions and unique mentions (i.e. unique entity names), as well as the proportion of each that are ambiguous, i.e. that belong to more than one entity type. Ambiguous mentions represent a small fraction of all unique entity names present in the training data, around 2%, but since these names tend to occur relatively frequently,[4] these ambiguous names make up a higher percentage of all mentions, upwards of 25% in the case of OntoNotes.

	CoNLL	**OntoNotes**
# mentions	23499	81828
# ambig. mentions	1632	20582
% ambig. mentions	6.9%	25.2%
# unique mentions	8082	25055
# ambig. unique mentions	132	576
% ambig. unique mentions	1.6%	2.3%

Table 1: Statistics on training sets. Ambiguous mentions are those that belong to more than one entity type in the training set.

		CoNLL	**OntoNotes**
Dev	# mentions	5942	11066
	# unseen mentions	1900	3511
	% unseen mentions	32.0%	31.7%
	# unique mentions	2809	4847
	# unseen unique mentions	1418	2707
	% unseen unique mentions	50.5%	55.8%
Test	# mentions	5648	11257
	# unseen mentions	2600	3597
	% unseen mentions	46.0%	32.0%
	# unique mentions	2637	4830
	# unseen unique mentions	1706	2659
	% unseen unique mentions	64.7%	55.1%

Table 2: Statistics on dev and test sets. Unseen mentions are those that do not appear in the corresponding training set.

Statistics on the development (or "dev") and test sets of CoNLL and OntoNotes are shown in Table 2. These include the number of mentions and unique names, as well as the proportion of each that are unseen in the corresponding training sets. As we can see, both benchmarks contain a high proportion of unseen mentions in the test set. Among the unique entity names in the test sets, 55-65% are not observed in the training data, which represents 32-46% of all test mentions in these benchmarks. It is worth noting here that, even when the training and test data are sampled from the same distribution, as in the case of CoNLL and OntoNotes, there is a significant rate of unseen mentions in the test set. This proportion depends on how the benchmark was constructed. It reaches 100% in the case of some benchmarks developed specifically to assess the capacity to detect new entity names (Derczynski et al., 2017).

Subset	**# Tokens**
BC	204K
BN	226K
MZ	198K
NW	489K
TC	104K
WB	170K

Table 3: Size of the 6 subsets of OntoNotes.

For cross-domain evaluation, we used the 6 subsets of OntoNotes corresponding to the 6 text sources, and evaluated using a leave-one-out setup, as we will explain in Section 4. We use the same train/dev/test splits[5] as in the 2012 CoNLL shared task on coreference (Pradhan et al., 2012), and exclude the pivot corpus, as in (Ghaddar and Langlais, 2018). As the sizes of the subsets we extracted do not quite match up with the numbers reported in the OntoNotes documentation (Weischedel et al., 2012, p. 6), we report the sizes we obtained in Table 3.

3. Our Proposal: `hardeval`

We propose an evaluation method called `hardeval`. The main idea of this method is to expose the training data (or some sample of it) and focus the evaluation on test tokens that represent specific sources of errors, namely unknown words and label shift (i.e. ambiguity).

3.1. Definition

`hardeval` computes the token error rate (TER) on various subsets of tokens in the test set. The two main subsets are unseen tokens, hereafter called `unseen`, and tokens whose label differs from what was observed during training, hereafter called `diff`. These subsets are identified using simple heuristics that are both system-agnostic and context-agnostic. For `unseen` tokens, we just check whether the token appeared in the training data. For `diff` tokens, we assume it appears in the training data at least once, and check whether the token's label is that word's most frequent label in the training data.
The `unseen` and `diff` tokens are further subdivided according to their label in the test set. Here is a description of these subsets, using the IO labeling format, where all words in a mention are labeled `I-<type>`, and all others `O`:

- **`unseen`** = test tokens that are not in the training set
  - `unseen-I`: label is `I-X` (`X` is any entity type)
  - `unseen-O`: label is `O`
- **`diff`** = test tokens whose label is not their most frequent label in the training set
  - `diff-I`: label was usually `O`, but is `I-X`
  - `diff-O`: label was usually `I-X`, but is `O`
  - `diff-E`: label was usually `I-X`, but is `I-Y` (different entity type)

[3]See `https://en.wikipedia.org/wiki/Inside-outside-beginning_(tagging)`

[4]If this is not immediately obvious, recall that a name that occurs only once can not be ambiguous, and that such low-frequency names make up a majority of all names.

[5]`http://conll.cemantix.org/2012/data.html`

Examples of each of these kinds of hard tokens are shown below in Section 3.2.

3.2. Deployment on CoNLL and OntoNotes

The sentence shown in Fig. 1 contains examples of both `unseen` and `diff` tokens in the CoNLL test set. It contains words that are not in the training data (i.e. *trans-Atlantic*, *Monopolies*, *Mergers*, and *complied*), a token that is part of a mention but usually is not (i.e. *and*), and a word that is usually part of a mention but belongs to different entity types (i.e. *British*).

> The [British]$_{\text{MISC}}$ government warned Friday that it would refer the proposed [trans-Atlantic]$_{\text{MISC}}$ alliance between [British Airways Plc]$_{\text{ORG}}$ **and** [American Airlines]$_{\text{ORG}}$ to [Britain]$_{\text{LOC}}$'s [Monopolies **and** Mergers Commission]$_{\text{ORG}}$ unless the carriers complied with a number of conditions .

Figure 1: A sentence from the CoNLL-2003 test set. Gold mentions are in brackets. Four `unseen` tokens are underlined; the first three are all `unseen-I` as they are part of a mention, and the last is `unseen-O`. Two occurrences of *and* are shown in bold; the second is `diff-I`. The token *American* and the second occurrence of *British* are both `diff-E`.

The two kinds of `unseen` tokens are easy to grasp, but the three different kinds of `diff` tokens may be a bit trickier at first. To see what kinds of tokens are `diff`, we inspected the `diff-I`, `diff-O`, and `diff-E` subsets identified using the the standard CoNLL train/test split. We can summarize our observations as follows:

`diff-I` The most frequent `diff-I` tokens in the CoNLL test set include *of*, *and*, and *I*, which are usually labeled O in the training set, but are sometimes part of a mention in the test set. For example, the vast majority of the occurrences of *and* are not part of a mention, but a few occurrences in the test set are, such as in the example shown in Fig. 1. In this example, we have two occurrences of the word *and*. The first is labeled O like the vast majority of occurrences seen in training, but the second is part of a mention, and is therefore part of the `diff-I` subset.

`diff-O` The five most frequent `diff-O` tokens in the CoNLL test set include four words which are almost always part of a MISC mention in the CoNLL training data, i.e. *DIVISION*, *LEAGUE*, *WESTERN*, and *EASTERN*, but are sometimes labeled O in the test set. One interesting case is that of the word *EASTERN*. In the CoNLL-2003 training set, this word appears 16 times, and is always part of the two-word sentence (heading) *EASTERN DIVISION*, which is consistently labeled as a mention of a MISC. However, in the test set, we find a total of six occurrences of the word *EASTERN*. Two of these appear in the heading *EASTERN DIVISION* as in the training set, and four others appear in the heading *EASTERN CONFERENCE*, an entity name that was not observed in training. For some reason, all of these are labeled O (not entity mentions). There may be an annotation consistency issue here, but in our experience, such issues are an unavoidable consequence of the practice of annotating data or collecting annotated data.

`diff-E` In the sentence shown in Fig. 1, the second occurrence of the word *British* is `diff-E`, as this token is usually part of a MISC mention in the training data, not an ORG as it is here. Likewise, the token *American* is `diff-E`, as it is usually a MISC as well. If we look at the most frequent `diff-E` tokens in the CoNLL test set, the top three are all words that appear most frequently as LOC in the training set, but sometimes appear as ORG in the test set, i.e. *United*, *Santa*, and *New*. If we focus on the word *Santa*, we observe that it appears three times in the CoNLL training set, within these mentions: *Santa Maria de Pocosol* (LOC), *Santa Barbara* (LOC), and *Santa Puglisi* (PER), so its most frequent type is LOC. In the test set, we can observe the appearance of an ORG-type mention that contains this word, that is *Santa Fe Pacific Gold Corp*. This name appears only once, but there are 18 occurrences of the shortened form *Santa Fe*, 16 of which were annotated as ORG-type mentions, and two of which were annotated as LOC-type mentions, but clearly refer to the aforementioned organization. In any case, most of the occurrences of *Santa* in the test set belong to an entity type which was never associated with that word in the training data.[6] This kind of label shift or ambiguity is especially frequent when there is a shift in the domain or source of the text, but as we can see, it also occurs in the single-distribution setting.

Table 4 shows the relative size of the `diff` and `unseen` token subsets identified in CoNLL and OntoNotes using the definitions in Section 3.1. The standard train/test splits were used to compute these subsets, but this was done for illustrative purposes only: for practical applications, we would use cross-validation, as explained later.

	# tokens (%)	
	CoNLL	**OntoNotes**
all	46435 (100.0)	152728 (100.0)
`unseen-I`	2537 (5.5)	1745 (1.1)
`unseen-O`	3119 (6.7)	1749 (1.1)
`unseen`	5656 (12.2)	3494 (2.3)
`diff-I`	201 (0.4)	3632 (2.4)
`diff-O`	215 (0.5)	1005 (0.7)
`diff-E`	676 (1.5)	2815 (1.8)
`diff`	1092 (2.4)	7452 (4.9)

Table 4: Number of `diff` and `unseen` tokens in test sets of CoNLL and OntoNotes, computed using the standard train/test split.

The results show that `unseen` tokens represent about 12% of test tokens in CoNLL, and 2% in OntoNotes, whereas `diff` tokens represent 2% and 5% of test tokens in CoNLL and OntoNotes respectively. Thus, `unseen` and `diff` tokens represent a small fraction of the test tokens, at least when we use the in-domain training set to identify them. Remember that the number of `unseen` or `diff` tokens depends on the training set that we provide to `hardeval`, but they do not depend on the system.

It is worth noting at this point that for tokens that are neither `unseen` nor `diff`, predicting their label mainly involves memorizing their most frequent label in the training

[6]There are also two mentions of the “person” Santa Claus.

set. This represents the vast majority of tokens in these benchmarks.

Comparing the number of `unseen` tokens shown in Table 4 to the number of unseen mentions shown in Table 2, one might wonder how the former can be lesser than the latter in the case of OntoNotes, whose test set contains 3597 unseen mentions, but 3494 `unseen` tokens. It is important to remember that the property of being unseen, i.e. not appearing in the training set, is evaluated at two different levels here: an entity name that never occurs in the training data – or does occur, but is never labeled as such – is considered an unseen mention, but it may contain words that did occur in the training data, and these would not be `diff` tokens by definition.

Observing the `unseen` and `diff` tokens in the CoNLL test set, as we have just done, makes it less useful as an estimator of generalization (assuming a fixed train/dev/test split), as we are gaining valuable information about the content of the test set, which could be used to artificially boost a system's performance on it. That being said, repeatedly using the same train/dev/test split to compare systems, as has happened repeatedly over the past 17 years in the case of CoNLL-2003, has accomplished the same, that is leaking information about the test set that renders it a biased estimator of generalization. The problem with fixed training/test splits has been highlighted by Gorman and Bedrick (2019), who recommend that NLP system comparisons be carried out by evaluating the systems on multiple, randomly generated training/test splits.[7]

It is important to remember we are exploring the test sets of these two commonly used benchmarks for illustrative purposes only. For practical applications, we would recommend using multiple, random training/test splits to identify `unseen` and `diff` tokens, as in k-fold cross-validation.

3.3. Properties of `hardeval`

Two properties of `hardeval` are worth noting:

- The `unseen` and `diff` subsets are disjoint, as are the five lower-level subsets. Tokens that are neither `unseen` nor `diff` are expected to be easier for a machine learning system to label, as this mainly involves memorization.
- The `diff-I`, `diff-O`, and `diff-E` subsets contain tokens that are likely to be false negatives, false positives, and type classification errors respectively. For instance, tokens in `diff-I` are usually not part of a mention in the training data, so a system trained on that data is more likely to fail to detect that they are part of a mention in the test set, which would generate false negatives. However, it is important to remember that the way in which these three subsets are identified is system-agnostic. That is, rather than restrict the evaluation to those predicted by a given system (as well as the gold-standard, human-annotated mentions), we restrict it to specific subsets of tokens that are likely errors, independent of the system.

[7]They also recommend using Bonferroni-corrected random split hypothesis testing to verify the significance of differences between systems.

In summary, `hardeval` computes the TER (i.e. percentage of mislabeled tokens) on `unseen` and `diff` tokens. The lower the score the better.

It is important to consider that a low TER on `diff-O` does not necessarily imply high-quality NER, as a system that always predicts O (and thus fails to ever detect any mentions) would have a TER of 0% on `diff-O`. So the TER on `diff-O` is not, on its own, a good metric to evaluate NER. Likewise for `unseen-O`. These metrics can provide useful insights, but should not be analyzed or optimized in isolation. In settings where a single evaluation metric is needed (e.g. to compare models or tune the hyperparameters of a particular model), we would recommend using the mean TER over the `diff` and `unseen` subsets, as explained below.

Lastly, it might be worth noting that we also implemented a stricter version of `diff`, whereby a token's label must never have been observed in training to qualify. This results in smaller subsets, and we prefer the looser definition.

4. Experiments

4.1. Methodology

We evaluated four NER systems using `hardeval` in both in-domain (ID) and out-of-domain (OOD) settings. The ID tests were conducted on the train/dev/test split of a single dataset, i.e. CoNLL or OntoNotes. For the OOD tests, we trained on the concatenated training sets of 5 of the 6 domains in OntoNotes, and evaluated on the test set of the held-out domain. The dev set, which was used for early stopping, was also OOD in this case.

In both ID and OOD settings, we computed the token error rate (TER) on `unseen` and `diff` tokens using the `hardeval` script, as well as the standard mention-level f-score, using the Perl evaluation script developed for the CoNLL-2003 shared task, called `conlleval`.

4.2. Systems Evaluated

We selected four systems for evaluation. They implement different approaches to representing tokens with features and labeling tokens based on those features:

- **Illinois** is the NER package in the CogComp-NLP toolkit.[8] It implements a rich set of hand-crafted features designed specifically for NER (Ratinov and Roth, 2009), such as lexical features (i.e. the tokens themselves), sub-word features (e.g. case and affix features), contextual features (e.g. the surrounding words, as well as non-local features), and knowledge-based features (e.g. gazetteers). It exploits a regularized averaged perceptron for sequence labeling.
- **NeuroNER** (Dernoncourt et al., 2017) is a neural NER toolkit.[9] It automatically extracts features using representation learning. Specifically, it employs a deep neural network in which BiLSTM layers learn representations at both character and word levels (with a single

[8]`https://github.com/CogComp/cogcomp-nlp`. We tested version 4.0.9.

[9]`https://github.com/Franck-Dernoncourt/NeuroNER`. We tested version 1.0.

BiLSTM layer at each level). These representations capture lexical, contextual, and sub-word features. No external knowledge or hand-crafted features are used. It exploits a CRF for sequence labeling.

- **spaCy** (Honnibal and Montani, 2017) is an NLP toolkit[10] that includes NER. It employs a neural network architecture in which convolution and attention layers automatically extract features at word level. Hand-crafted sub-word features are injected separately into the network. The model exploits a transition-based algorithm (i.e. stack-LSTM) for sequence labeling.
- **BERT** (Devlin et al., 2019) is an algorithm that employs self-supervised language model pre-training of a transformer architecture (Vaswani et al., 2017), followed by supervised fine-tuning on a given target task.[11] We used the pre-trained model named "bert-large-cased-whole-word-masking", which was pre-trained on a large corpus of text using whole-word masking for the masked language model.[12] The model learns to capture contextual features through language modeling and supervised fine-tuning. No external knowledge is used, apart from the large, unannotated text corpus used for pre-training. Sub-word features are obtained by splitting the input tokens into sub-word units (Sennrich et al., 2016).

For each of these systems, we used the default or recommended configuration.[13] When forced to select a feature or hyperparameter setting, we did a minimum of optimization to make sure we achieved scores close to those reported by the developers.[14] Otherwise, we treated the systems as black boxes.

5. Results

5.1. Results: `conlleval`

F-scores achieved by the four systems in the ID setting are shown in Table 5. Also shown are the current state-of-the-art (SOTA) scores on these two datasets according to Li et al. (2019). These results show that modern NER systems achieve f-scores upwards of 90% on CoNLL and OntoNotes in the ID setting. Thus, the standard evaluation procedure for NER suggests that modern systems achieve very high performance, and that there is little room for improvement (RFI). This RFI represents around 10% of the mentions that were detected by either the human annotators or a given system (or both).

	CoNLL	**OntoNotes**
Illinois	0.907	0.836
Neuro	0.899	0.866
spaCy	0.881	0.850
BERT	0.916	0.894
SOTA (Li et al., 2019)	0.930	0.911

Table 5: Mention-level f-scores. In-domain training. SOTA denotes current state-of-the-art.

Regarding the system rankings, the results show that Illinois performs well on CoNLL, but neural systems perform better on OntoNotes, which is a richer, more varied, and more challenging dataset.

F-scores achieved in the OOD setting are shown in Table 6. The scores on the various subsets of OntoNotes are all inferior to the ID results achieved on the complete dataset, often by a large margin. Note that there is 1/6 less training data for each test in the OOD setting, which may explain part of the drop in performance. On the most challenging subset, f-scores are just above 75%. Averaged across the 6 subsets, the f-scores are about 6-8 points lower than the overall f-score obtained on the complete dataset in the ID setting,[15] which suggests significantly more RFI.

	BC	**BN**	**MZ**	**NW**	**TC**	**WB**	**avg**
Illinois	0.76	0.83	0.77	0.76	0.68	0.74	0.76
Neuro	0.79	0.85	0.79	0.81	0.71	0.75	0.78
spaCy	0.79	0.84	0.75	0.78	0.71	0.76	0.77
BERT	0.84	0.89	0.87	0.86	0.75	0.80	0.84

Table 6: Mention-level f-scores on the 6 subsets of OntoNotes. Out-of-domain training.

5.2. Results: `hardeval`

Tables 7 and 8 show the token error rates of the systems on CoNLL and OntoNotes respectively, using ID training. For reference, the overall TER on all tokens is around 2-3% in all cases. The results of `hardeval` show that modern NER systems achieve a TER around 6-9% on `unseen` tokens in CoNLL, and 8-16% in OntoNotes. On `diff` tokens, error rates are much higher: around 27-40% on CoNLL and 22-41% on OntoNotes. This suggests label shift or ambiguity is more challenging for these systems than `unseen` tokens, at least in terms of TER, and that there is a lot of RFI here. Also note that the basis for the RFI is defined in a way that is system-agnostic, which is not the case if we use the standard evaluation method, as we highlighted earlier.

It might also be worth noting that the TER is higher on `unseen-I` than `unseen-O`, which indicates it is harder for NER systems to correctly label unseen tokens that are

[10]`https://spaCy.io/`. We tested version 2.0.11.

[11]We use the library by HuggingFace (Wolf et al., 2019). See `https://github.com/huggingface/transformers`. We tested version 2.1.1.

[12]See `https://github.com/google-research/bert` for more info on how this model was pre-trained.

[13]Installation scripts are provided in our code repository.

[14]This was the case for the word embedding table in the case of spaCy. We ended up using the glove.840B.300d embeddings (`https://nlp.stanford.edu/projects/glove/`), with 200K unique embeddings after pruning the vocabulary. In the case of NeuroNER, we used the glove.6B.100d embeddings as recommended.

[15]Recall that these 2 sets of results are not directly comparable as we do not average over the 6 subsets in the ID setting. However, since the sizes of the 6 subsets do not vary a lot, it is safe to assume they are roughly comparable. This comparison seems more fair than doing ID training on the 6 subsets and averaging, as we would use 4/5 less training data for each of the ID tests than for the OOD tests.

	`unseen`			`diff`			
	all	I	O	all	I	O	E
Illinois	0.07	0.12	0.03	0.34	0.41	0.42	0.29
Neuro	0.08	0.12	0.05	0.34	0.38	0.50	0.28
spaCy	0.09	0.15	0.05	0.40	0.48	0.52	0.34
BERT	0.06	0.09	0.03	0.27	0.29	0.50	0.18

Table 7: Token error rates on CoNLL. In-domain training.

	`unseen`			`diff`			
	all	I	O	all	I	O	E
Illinois	0.16	0.29	0.03	0.41	0.46	0.40	0.36
Neuro	0.13	0.22	0.03	0.27	0.24	0.42	0.25
spaCy	0.15	0.26	0.05	0.33	0.30	0.50	0.30
BERT	0.08	0.13	0.03	0.22	0.21	0.37	0.18

Table 8: Token error rates on OntoNotes. In-domain training.

part of a mention than those that are not, as we might expect.
Like OOD evaluation, evaluating the TER on `diff` tokens paints a less rosy picture of NER performance than the standard evaluation method. And like OOD evaluation, it gives us an estimate of robustness to shifts in the data distribution, while being a simpler method, since it does not require us to look at a variety of datasets, as has sometimes been proposed to assess robustness – see Section 7.
`hardeval` can also be used in a cross-domain evaluation setting. Table 9 shows the OOD error rates of the four systems; for conciseness, this table only shows the average TER over the 6 subsets of OntoNotes (as in the last column of Table 6). If we compare these to the ID error rates on OntoNotes, we see that the TER is higher on both `unseen` (in particular `unseen-I`) and `diff` tokens in the OOD setting. This is likely due to a greater shift between training and test data. It may also be partly due to the fact that we use 1/6 less training data in the OOD case, as we mentioned earlier.

	`unseen`			`diff`			
	all	I	O	all	I	O	E
Illinois	0.16	0.34	0.03	0.52	0.59	0.39	0.49
Neuro	0.15	0.30	0.05	0.41	0.40	0.46	0.41
spaCy	0.16	0.32	0.04	0.45	0.42	0.53	0.46
BERT	0.09	0.20	0.03	0.32	0.32	0.37	0.31

Table 9: Token error rates on OntoNotes (average over 6 subsets). Out-of-domain training.

A detailed breakdown of the ID error rates on CoNLL and OntoNotes is shown in Table 10. These results show that `unseen` and `diff` tokens explain the majority of errors made by these systems on both CoNLL and OntoNotes, although these subsets represent only a small fraction of the test tokens, as we showed previously (see Table 4).
To assess the performance of NER systems on both `unseen` and `diff` using a single metric, we propose to use the average of the TER on those two disjoint subsets. Table 11 summarizes the `hardeval` scores (mean TER on `unseen` and `diff` tokens) of the four systems on CoNLL and OntoNotes, in both ID and OOD settings in the latter

	CoNLL			**OntoNotes**		
	unseen	diff	other	unseen	diff	other
Illinois	40.0	48.2	11.8	11.1	62.0	26.9
Neuro	42.1	42.8	15.1	11.4	51.6	37.0
spaCy	42.9	44.6	12.5	12.2	55.5	32.3
BERT	40.0	34.2	25.8	9.2	55.5	35.3

Table 10: Error breakdown (% of mislabeled tokens). In-domain training.

case. It may be worth noting that this metric indicates that three of the four systems tested are actually doing better on OntoNotes than CoNLL, at least in the in-domain setting. At any rate, we argue that it gives us a more meaningful idea of the room for improvement, on a system-agnostic basis. To obtain a more fine-grained view of the remaining challenges, one can look separately at the TER on `diff` and `unseen` tokens, or go even deeper by looking at the test tokens that are in these subsets, as we did in Section 3.2.[16]

	CoNLL	**OntoNotes**	
		ID	OOD
Illinois	0.206	0.285	0.338
Neuro	0.212	0.198	0.281
spaCy	0.247	0.239	0.307
BERT	0.163	0.149	0.207

Table 11: `hardeval` score (mean TER on `unseen` and `diff`) on CoNLL and OntoNotes. Out-of-domain results on OntoNotes (averaged over its 6 subsets) are shown in the last column.

6. Discussion

Many NLP and machine learning researchers still strive to beat the state-of-the-art on NER benchmarks such as CoNLL or OntoNotes. Given the limitations of the standard evaluation paradigm, it is hard to tell:

1. where a gain in f-score might come from, e.g. a better model, better optimization or better data.
2. what such gains actually mean: does a small increase in f-score really mean the model is better at learning what a named entity is? Or is it just better at modeling statistical noise, or even annotation errors?

Regarding the first problem, let us note that a fair comparison between NER systems should account for the fact that the systems may have access to different resources, aside from the training data: knowledge-based resources, pre-trained word embeddings, pre-trained language models, etc. This was the case for the systems we evaluated, as we explained in Section 4.2. As for the training data, in this work, we restricted the training data available to the systems, and that data was then used by `hardeval`, along with the test set, to identify the `unseen` and `diff` subsets used for evaluation. This provides a degree of fairness,

[16]It is important to keep in mind the caveat we expressed regarding the repeated use of the same train/test splits.

compared to an evaluation which only considers the test set and does not constrain the training data.

Let us also repeat that we conducted a black box evaluation, and relied on the default or recommended settings of the four systems, so the results shown in this paper depend on how effectively these four systems were tuned. It is also important to remember that the systems that we tested continue to evolve, and the current versions may perform better than those we tested.

Regarding the second problem, `hardeval` allows us to quantify the RFI in a way which is system-agnostic and distinguishes specific sources of errors. A more fine-grained analysis can be conducted by inspecting the errors made by a given system on the `unseen` and `diff` subsets. Inspecting the tokens that make up the `unseen` and `diff` subsets may also reveal annotation inconsistencies, as we showed in Section 3.2. However, if such inconsistencies are not eliminated from the dataset, they remain a potential source of bias for any evaluation metric, including those proposed in this paper.

It is worth noting that the `hardeval` evaluation method can be applied to any (partially or completely) supervised sequence labeling task where unseen tokens and label shift are likely to occur. The simple heuristics we use to identify hard subsets of tokens might also be used to improve the performance of NER on these subsets, using a framework such as slice-based learning (Chen et al., 2019). For such purposes, the `diff` tokens in the training set could be identified by cross-validation, as mentioned in Section 3.2.

7. Related Work

Many recent works raise issues with standard evaluation methods in NLP, such as:

- Ethical issues concerning shared tasks in NLP (Parra Escartín et al., 2017)
- Problems arising from the repeated use of standard train/dev/test splits (Gorman and Bedrick, 2019)
- Whether high performance on a dataset actually means high performance on the task at hand, or whether it just means better modeling of annotator idiosyncrasies (Geva et al., 2019)
- How leaderboards have become meaningless now that large neural language models pre-trained on large text corpora have become standard for many NLP tasks (Rogers, 2019)
- The fact that standard evaluation methods completely disregard the cost of training and tuning models, in terms of energy, money, and ecological impact (Schwartz et al., 2019)

Many of these issues are part of a growing discussion around the failure of machine learning and NLP models to perform in real-world settings, because of a lack of robustness. Tools to better assess robustness are therefore required to improve ML and NLP methods.

> "One of the challenges of robustness is that it is hard to study systematically. How do we benchmark how well an algorithm trained on one distribution performs on a different distribution? Performance on brand-new data seems to involve a huge component of luck. That's why the amount of academic work on robustness is significantly smaller than its practical importance. Better benchmarks will help drive academic research." (Ng, 2019).

To better assess robustness, we need better evaluation methods. In that spirit, we hope the method presented here will enable researchers to develop more robust models.

Regarding the issue of robustness, it is worth noting that out-of-domain and cross-domain evaluations of NER systems have been carried out in a few studies (Augenstein et al., 2017; Agerri and Rigau, 2017; Ghaddar and Langlais, 2018; Taillé et al., 2020). We believe out-of-domain evaluation provides valuable information, but comparing results across a variety of annotated corpora is costly and not always feasible, and summarizing their results can be tricky. So we would argue there is a need for an evaluation method that focuses on robustness but can be used even with a single dataset, such as the one presented here.

It is worth noting that Augenstein et al. (2017) also looked at the accuracy of NER systems on unseen mentions to better assess their capacity to generalize, as did Taillé et al. (2020), who further subdivided these mentions into partial matches (which contain at least one token which was seen in a mention of the same type, excluding stop words) and completely new mentions. Finally, let us repeat that there are datasets that have been designed specifically to evaluate NER systems on unseen mentions (Derczynski et al., 2017).

8. Conclusion

To assess the robustness of NER systems, we propose an evaluation method that focuses on subsets of tokens that represent specific sources of errors: unknown words and label shift or ambiguity. These subsets, which we call `unseen` and `diff`, provide a system-agnostic basis for evaluating specific sources of NER errors and assessing room for improvement in terms of robustness.

In this paper, we analyzed the `unseen` and `diff` tokens in two widely-used NER benchmarks, then we conducted a black-box evaluation of various approaches to NER based on `unseen` and `diff` tokens. Results show that `unseen` and `diff` tokens explain the majority of errors made by modern NER systems, although these subsets represent only a small fraction of the test tokens. They also indicate that label shift is harder to deal with than unknown words, and that there is much more room for improvement than the standard NER evaluation procedure would suggest.

Future work might look at incorporating the metrics used in this paper into a single metric that evaluates all tokens, including those that are likely easy to memorize, weighted by their difficulty. It would also be interesting to try to figure out what tokens tend to be challenging aside from `diff` and `unseen`. This might involve looking at the context of a given token, not just the token, its label, and the labels that were observed previously for that word. We are also interested in exploiting the analysis of `unseen` and `diff` tokens in training data to train more robust models.

9. Acknowledgements

The authors thank IVADO and Irosoft for supporting this research and the anonymous reviewers for their helpful comments.

10. Bibliographical References

Agerri, R. and Rigau, G. (2017). Robust multilingual named entity recognition with shallow semi-supervised features. *ArXiv preprint:1701.09123*.

Augenstein, I., Derczynski, L., and Bontcheva, K. (2017). Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61 – 83.

Chen, V. S., Wu, S., Weng, Z., Ratner, A., and Ré, C. (2019). Slice-based learning: A programming model for residual learning in critical data slices. *ArXiv preprint:1909.06349*.

Derczynski, L., Nichols, E., van Erp, M., and Limsopatham, N. (2017). Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

Dernoncourt, F., Lee, J. Y., and Szolovits, P. (2017). NeuroNER: An easy-to-use program for named-entity recognition based on neural networks. *ArXiv preprint:1705.05487*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Geva, M., Goldberg, Y., and Berant, J. (2019). Are we modeling the task or the annotator? An investigation of annotator bias in natural language understanding datasets. *ArXiv preprint:1908.07898*.

Ghaddar, A. and Langlais, P. (2018). Robust lexical features for improved neural network named-entity recognition. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1896–1907. ACL.

Gorman, K. and Bedrick, S. (2019). We need to talk about standard splits. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2786–2791, Florence, Italy, July. Association for Computational Linguistics.

Honnibal, M. and Montani, I. (2017). spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., and Weischedel, R. (2006). OntoNotes: the 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL (Short Papers)*, pages 57–60. ACL.

Li, X., Feng, J., Meng, Y., Han, Q., Wu, F., and Li, J. (2019). A unified MRC framework for named entity recognition. *ArXiv preprint:1910.11476*.

Ng, A. (2019). The Batch, November 6, 2019. https://info.deeplearning.ai/the-batch-deepmind-masters-starcraft-2-ai-attacks-on-amazon-a-career-in-robot-management-banks-embrace-bots.

Parra Escartín, C., Reijers, W., Lynn, T., Moorkens, J., Way, A., and Liu, C.-H. (2017). Ethical considerations in NLP shared tasks. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 66–73, Valencia, Spain, April. Association for Computational Linguistics.

Pradhan, S., Moschitti, A., Xue, N., Uryupina, O., and Zhang, Y. (2012). CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proceedings of the Joint Conference on EMNLP and CoNLL: Shared Task*, pages 1–40. ACL.

Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. ACL.

Rogers, A. (2019). How the Transformers broke NLP leaderboards. https://hackingsemantics.xyz/2019/leaderboards/.

Schwartz, R., Dodge, J., Smith, N. A., and Etzioni, O. (2019). Green AI. *ArXiv preprint:1907.10597*.

Sennrich, R., Haddow, B., and Birch, A. (2016). Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August. Association for Computational Linguistics.

Taillé, B., Guigue, V., and Gallinari, P. (2020). Contextualized embeddings in named-entity recognition: An empirical study on generalization. *ArXiv preprint:2001.08053*.

Tjong Kim Sang, E. F. and De Meulder, F. (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 (Volume 4)*, pages 142–147. ACL.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Weischedel, R., Pradhan, S., Ramshaw, L., Kaufman, J., Franchini, M., El-Bachouti, M., Xue, N., Palmer, M., Hwang, J. D., Bonial, C., Choi, J., Mansouri, A., Foster, M., aati Hawwary, A., Marcus, M., Taylor, A., Greenberg, C., Hovy, E., Belvin, R., and Houston, A. (2012). Ontonotes Release 5.0 with OntoNotes DB Tool v0.999 beta. Technical report, LDC.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J. (2019). Huggingface's transformers: State-of-the-art natural language processing. *ArXiv preprint:1910.03771*.