

A Replicable Comparison Study of NER Software: StanfordNLP, NLTK, OpenNLP, SpaCy, Gate

Xavier Schmitt*, Sylvain Kubler[†], Jérémy Robert*, Mike Papadakis*, Yves LeTraon*

**SnT*

University of Luxembourg
Luxembourg, Luxembourg

Email: Xavier.Schmitt.001@student.uni.lu, {Jeremy.Robert, Mike.Papadakis, Yves.LeTraon}@uni.lu

[†]*CRAN*

University of Lorraine
Nancy, France

Email: S.Kubler@univ-lorraine.fr

Abstract—Named Entity Recognition (NER) is a key building block of any Natural Language Processing (NLP) system, making possible the detection and classification of entities (e.g., Person, Location) in any given text. While a large number of NER software exist today, it remains difficult for NLP and NER practitioners to clearly and objectively identify what software perform(s) the best. One of the reasons is the difference in results across the literature and the lack of information needed to be able to fully reproduce the experiment. To overcome this problem, this paper presents a comprehensive and replicable study to assess the performance of NER software, thus laying the groundwork for future benchmarking and meaningful comparison studies. As part of our experiments, the latest version of five well-known NER software were selected, along with two distinct corpora. We observe a discrepancy between the result we get and the result found in the literature being around 50% in certain cases. We also found that StanfordNLP usually performs the best.

Index Terms—Information Extraction, Methodology, Named Entity Recognition

I. INTRODUCTION

Natural Language Processing (NLP) is a branch of artificial intelligence that makes it possible for computers to understand, process and generate language just as people do. Information extraction is an important step of any NLP system, as it helps to automatically extract structured information from machine readable documents [1]. It is used for various purposes such as knowledge extraction, web scraping, text mining, and so forth.

Named Entity Recognition (NER) plays a key role in information extraction, allowing for the identification of “entities” (e.g., Person, Location). NER is widely used in machine translation, question answering information retrieval, and automatic summarization [2]. Originally, Person, Organization, Location were the first three entities considered for semantically classifying words. In the following years, new entities were defined to meet domain-specific needs (e.g., in the medical or legal sectors) [3], [4]. Various criteria must be taken into consideration before selecting and using a NER software such as its performance, cost, documentation, license, *etc.* It is nonetheless difficult to find comprehensive state-of-the-art comparison studies, as each software is often evaluated

in an independent manner, based on different corpora. Even when a set of corpus is considered, the underlying evaluation methodology and experimental conditions may differ from one study to another, or may simply not be sufficiently detailed. This makes it difficult to fairly compare the performance of different NER software, but also prevents scholars and practitioners to replicate the experiments.

This paper will provide the results of the latest version of five popular NER software (StanfordNLP, NLTK, OpenNLP, SpaCy and Gate) tested on a well known corpus (CoNLL2003) and a more recent one (GMB). Our methodology will be presented in order to give the possibility to reproduce the experiments. Using our reproducible methodology, our results show that StanfordNLP performs between 15% and 30% better on selected corpora than the other software tested. However, we were not able to retrieve the same results as the ones we can find in the literature for every tested software. The difference observed can be up to 66%.

Evidence that existing state-of-the-art studies lack of information for experiment reproducibility purposes and differences in results is discussed in section II. A comprehensive and replicable study that ensures a fair and meaningful evaluation and comparison of NER software is then presented in section III. Five NER software (NLTK, OpenNLP, StanfordNLP, SpaCy, Gate) are compared in section IV. Beyond discussing results and findings of the conducted experiment, similarity (and non-similarity) with state-of-the-art results is also discussed, surprisingly showing that significant differences exist. Conclusion is given in section V.

II. NER SOFTWARE EVALUATION

Section II-A briefly discusses the main tasks/stages that compose any given NER software. Section II-B provides a more in-depth analysis of existing evaluation studies and the extent to which they fail to be fully transparent to allow proper reproducibility of results and experiments.

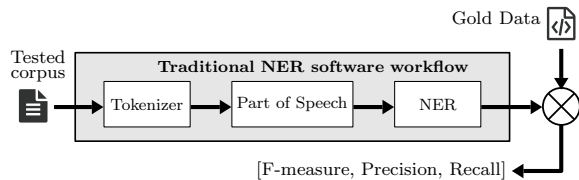


Fig. 1. Traditional evaluation methodology of NER software performance

A. NER: Concepts & Functioning

NER aims to identify and semantically classify words/entities from a text. A traditional NER software consists of three main tasks, as summarized in Fig. 1.

The way to evaluate the performance of any given NER software often follows the workflow depicted in Fig. 1. As inputs, both a corpus of reference and the associated “gold data” are used in order to evaluate the extent to which the identification/classification resulting from the NER software matches with the gold data.

B. NER Software Evaluation

Various studies have been carried out in the literature to determine which NER software performs the best. TABLE I gives an overview of the most striking evaluation studies in this domain, in which information about the corpus, software, and metrics considered for evaluation is reported. Furthermore, we report the extent to which these studies provide information needed to replicate the evaluation experiments, which includes the type of classifier based on which the evaluated software has been trained on, and the software version used for the corresponding experiment.

1) *NER Software*: From the set of evaluated NER software, it can first be noted that NER software are either developed for being used in a specific domain (e.g., social media, healthcare) or in a generic manner (i.e., being domain-independent). StanfordNLP¹, NLTK, SpaCy and OpenNLP are, to date, the most well-known software for generic use, each one having specific features and tuning. StanfordNLP [5] is a JAVA toolkit that provides a broad range of tools like PoS, NER tagger, *etc.* [6]. SpaCy is known for its rapidity in parsing [7], while NLTK offers a wide range of libraries and modules for symbolic and statistical NLP purposes [8].

2) *Corpus*: As was discussed in section II-A, the choice of the corpus is an important step in the performance evaluation process. Many annotated corpora exist, some of them having been specifically created for scholars such as *CoNLL 2003*, *MUC-6*, *MUC-7* using newswire [9]–[11], or *ACE 2005* [12] using weblogs, broadcast news, newsgroups and broadcast conversations. Other corpora were built over time to extend, or diversify, the nature of already existing corpora such as *OntoNotes*, *WikiGold* [13]; or more specialized corpora such as *Ritter twitter* [14] and *UMBC* [15].

¹In some studies, readers may see StanfordNER, which is actually a subpart of StanfordNLP. However, for consistency purposes, StanfordNLP will be used in this paper.

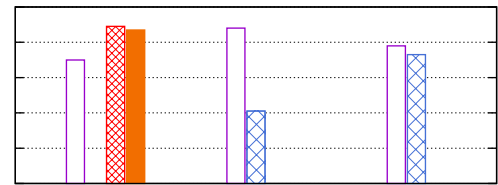


Fig. 2. Divergence of results in the literature with regard to the *StanfordNLP* software. Although, with *CoNLL 2003*, the max divergence is about 15%, it appears to be unmistakable when looking at the *Ritter* corpus (the F1-score of [16] being more than double the one of [17]).

3) *Motivation*: One important point about the reported evaluation studies is that they often lead to difference in results, sometimes in a substantial way, when evaluating a same software. Let us mention, for example, the study of [16] in which 7 NER software are compared (*cf.*, TABLE I). The authors conclude that NLTK and OpenNLP have similar results, and most importantly better than StanfordNLP. However, one may wonder why the results of StanfordNLP are that low considering that the official StanfordNLP website² announces much better results (note: both studies having used the same corpus for evaluation purposes). Such divergence of results is emphasized in Fig. 2, where the F1-score obtained by 4 out of the 6 evaluation studies presented in TABLE II-B have been reported, which all evaluate a same software (StanfordNLP in this case) based on a common set of corpora (*CoNLL 2003*, *Ritter*, *MSM2013*). The result is unmistakable, especially considering the *Ritter* corpus, as the F1-score of [16] is more than double the one of [17].

It should also be noted that existing evaluation studies often fail to provide all the necessary information to allow the reproducibility of experiments, resulting in the impossibility to obtain similar results for comparison purposes. For example, looking at the 6 evaluation studies reported in TABLE I, only 24% of the evaluated NER software (i.e., 6 out of 25) provide information about the type of classifier used, while only 16% detail the version of the software used for the experiment. This lack of information, and the impossibility to replicate the evaluation, thus make it very difficult to replicate the experiments, and most importantly to be able to judge the quality and completeness of the results. To overcome this problem, we proceeded in a two-stage fashion. First, we contacted the authors of the reported studies in order to request for the missing information; so far, only Pinto et al. provided new information, which is highlighted in bold in TABLE I. Second, we propose and present a clear and replicable comparison study in the next section.

²<https://nlp.stanford.edu/projects/project-ner.shtml>

Reference	Corpus	Software	License	Classifier	Version
[16]	CoNLL 2003 Ritter MSM2013	OpenNLP	Apache Software Lic.	N/A	N/A
		StanfordNLP	GNU GPL	CoNLL, ACE, MUC	3.6.0
		NLTK	Apache Lic. v2	ACE	N/A
		Pattern	BSD	N/A	N/A
		TweetNLP	GPL v2	N/A	N/A
		TweeterNLP	GNU GPL	N/A	N/A
[21]	Wikigold	TwitIE	N/A	N/A	N/A
		SpaCy	MIT License	N/A	N/A
		StanfordNER	GNU GPL	CoNLL MUC6/7, ACE	v3.6
		NLTK	Apache Lic. v2	N/A	N/A
[17]	MSM2013 Ritter UMBC	Alias-i LingPipe	Royalty Free Lic. v1	MUC6	4.1
		Annie (Gate)	GPL v3	Gazetter	Gate v8
		StanfordNER	GNU GPL	CoNLL, ACE	N/A
		DBpedia Spotlight	Apache Lic. v2	N/A	N/A
		Lupedia	N/A	N/A	N/A
		Ritter et al.	GPL v3	N/A	N/A
		Alchemy API	Non Commercial	N/A	N/A
		NERD-ML	GPL v3	N/A	N/A
		YODIE	N/A	N/A	N/A
		Zemanta	Non Commercial	N/A	N/A
[20]	CoNLL 2003	TextRazor	Non Commercial	N/A	N/A
		StanfordNLP	GNU GPL	N/A	N/A
[22]	OntoNotes 5	Annie (Gate)	GPL v3	N/A	N/A
		SpaCy	MIT License	Small, Medium, Large OntoNotes 5	v2.x
[23]	CoNLL 2003	StanfordNLP	GNU GPL	N/A	N/A

TABLE I

EXPERIMENTAL CONDITIONS OF NER SOFTWARE EVALUATION STUDIES IN THE LITERATURE. THIS TABLE SHOWS THE LACK OF INFORMATION TO BE ABLE TO REPRODUCE THE EXPERIMENTS. AS COMPLEMENTARY INFORMATION, FIG. 2 SHOWS THE DIVERGENCE IN RESULTS FOUND IN THE LITERATURE (FROM 15% TO 50%), WHICH MOTIVATES US IN PROPOSING A REPLICABLE COMPARISON STUDY OF NER SOFTWARE.

III. NER SOFTWARE COMPARISON STUDY

An overall overview of the proposed comparison study³ is given in Fig. 3, which consists of the selection of a corpus (and associated gold data), of the set of NER software to be evaluated, as well as of the set of metrics. These steps are respectively described in sections III-A, III-B and III-C.

A. Corpus Selection

A set of criteria has been defined for selecting the corpus used for evaluation purposes, namely that the corpus should not be domain-specific, should be freely available and in English. Given these criteria, we selected two distinct corpora: (i) *Groningen Meaning Bank*⁴: consists of newswire texts and a collection of texts from the open ANC and Aesop's fables [18]; (ii) *Reuters Corpus that was used in the shared task of the CoNLL conference 2003*⁵: consists of extracts of Reuters articles.

B. NER Software Selection

Since the goal of our study is to carry out a comparison study of NER software, criteria for selecting the set of software that are going to be evaluated have been defined, which include the fact that software must be: i) free of charge; ii) for unlimited use and with available documentation; iii) available under

Linux environment; and iv) able to recognize at least the three following entities: Person (PER); Organization (ORG); Location (LOC). Given these criteria, the following five NER software were selected: StanfordNLP, NLTK, OpenNLP, SpaCy, Gate, whose results and findings are discussed in section IV.

C. Performance Evaluation Methodology

In our study, the performance of a given NER software is evaluated by comparing the set of entity tags identified by the NER software with the ones that have been specified by the gold data. The overall evaluation process is depicted in Fig. 3. Output of each NER software has been converted to the same Inside-Outside-Beginning (IOB) tagging used by the CoNLL 2003 corpus [19].

It should be noted that each software comes along with its own features, including its own tag representation, programming language and format. It is therefore needed to standardize each software's output to be able to use the same evaluation script. As detailed in section III-B, all selected tools recognize at least the LOC, PER, ORG tags, as GMB and CoNLL 2003's gold data does. All other tags identified by one or more NER software are mapped to O (Other). Only one exception was made for SpaCy and NLTK, as both use the Geopolitical Entity (GPE) tagging for representing states, countries, cities, which have thereby been mapped to the LOC entity.

Regarding the scoring technique, several techniques exist, as the ones proposed at the MUC and CoNLL conferences [9], [10]. In this study, the CoNLL 2003 shared task scoring

³GitHub page of the source code: <https://github.com/xavierschmitt/NEREvaluationScriptPreparation>

⁴All files being freely available at: <http://gmb.let.rug.nl/>

⁵All files being freely available at NIST: <https://trec.nist.gov/data/reuters/reuters.html>

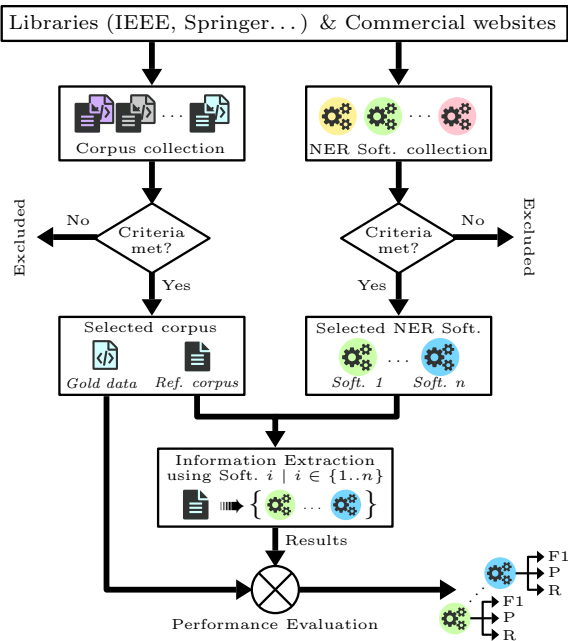


Fig. 3. Proposed workflow for evaluation

Algorithm 1: Appended Gold Data**Input** : \mathcal{G} of size k , \mathcal{S} of size l **Output**: \mathcal{G}_{res}

- 1 $\mathcal{G}_{res} = \mathcal{G}$; for $i \leftarrow 1$ to k do
- 2 $res = find(chunk[\mathcal{G}(i)] == chunk[\mathcal{S}(k)]);$
- 3 $\mathcal{G}_{res}(i) \leftarrow \mathcal{G}_{res}(i) || res;$

protocol has been used, which does proceed to an exact matching – *between the gold data and software outcome* – of both the chunk (specifically the chunk boundaries) and the tag. Algorithm 1 details how such a protocol has been implemented, where inputs \mathcal{G} and \mathcal{S} respectively refer to the Gold Data (consisting of k lines/chunks) and to the software output (consisting of l lines/chunks), while \mathcal{G}_{res} corresponds to the gold data which is appended with the software results⁶. Let us note that a chunk from \mathcal{G} is either uniquely identified in \mathcal{S} or considered as null/empty. To evaluate the performance of NER software, P (Precision), R (Recall) and F (F-measure) metrics are considered (*cf.*, Fig. 3), which are the most commonly used in the NLP community.

IV. PERFORMANCE EVALUATION STUDY

The comparison study has been carried out considering the five selected NER software. TABLE II provides insight into the experimental conditions of our experiment, detailing the software version, implemented algorithm and classifiers, as well as the underlying programming language. Section IV-A discusses results and findings obtained from the experiment.

⁶“||” in Algorithm 1 refers to the concatenation symbol.

Name	Prog.	Vers.	Algo	Classifiers
Stanford NLP	Java	3.9.2	CRF	CoNLL, MUC6-7, ACE
OpenNLP	Java	1.9	Max. entropy	OntoNotes
SpaCy	Python	2.0.16	Neural (2.0)	Stanford NER
NLTK	Python	3.4	Max entropy	
GATE	Java	8.5.1	JAPE	

TABLE II
NER SOFTWARE SELECTED FOR EVALUATION

A. Results & Findings

TABLE III provides an overview of the performance evaluation results obtained for each software⁷ with regard to the LOC, PER and ORG tags, as well as from an Overall viewpoint (i.e., taking all tag categories into account). Regarding CoNLL 2003 corpus, one can observe that StanfordNLP is clearly outperforming the other software, whether the entity category or the metric (P, R, F1). More interestingly, OpenNLP, NLTK, Gate and SpaCy get significantly much lower results than StanfordNLP at identifying ORG entities, while the gap is not that significant when looking at the LOC and PER entities. From an Overall viewpoint, we can state that OpenNLP, NLTK, Gate and SpaCy have similar results, except for OpenNLP that has a poor Recall score compared with the three other software while having the highest Precision. Regarding GMB corpus, StanfordNLP is still outperforming the other software. But we can clearly notice that the disparity is not as significant as it is with CoNLL 2003, approximately twice less. NLTK has even managed to be slightly better at tagging Location than StanfordNLP. Compared to CoNLL 2003, results are stable for SpaCy and Gate, better for NLTK and worst for StanfordNLP and OpenNLP. The main reason is that StanfordNLP comes with a classifier by default that has been trained partially on CoNLL 2003, where other software were not. This is why our experiments on GMB should provide results closer to what you can expect with your own corpus.

TABLE IV provides an overview of the “Overall” results obtained (*cf.*, last column) compared with the state-of-the-art studies introduced in TABLE I. A color code is used (*cf.*, table’s legend) in order to provide an at-a-glance view of the extent to which our results are (or not) similar to the ones obtained from the literature. While our results are quite close with studies having evaluated StanfordNLP (difference of between 0 and 20%), the difference is surprisingly more significant regarding the three other software (i.e., NLTK, Gate and OpenNLP), even reaching a difference of 66% with OpenNLP compared with the study of [16]. The reason for such differences is likely due to the fact that the experimental setting between our study and the others is different (e.g., difference in the software version, classifier, *etc.*). However, once more, the lack of information about the exact experimental conditions makes it difficult to confidently state that this is the primary reason behind the difference in results.

⁷Default parameters of each software were used.

Software	Entity	CoNLL 2003			GMB		
		P	R	F1	P	R	F1
Stanford NLP	LOC	91.30	88.73	90	83.10	63.64	72.08
	ORG	86.32	80.92	83.53	71.40	47.42	56.99
	PER	92.72	82.68	87.41	78.59	84.70	81.53
	Overall	90.06	73.67	81.05	79.81	63.74	70.88
NLTK	LOC	52.47	65.47	58.26	77.13	77.1	77.12
	ORG	36.20	24.80	29.44	42.06	35.54	38.53
	PER	61.09	66.11	63.50	38.07	55.87	45.28
	Overall	51.78	45.56	48.47	60.96	63.91	62.40
Gate	LOC	59.63	78.63	67.82	79.03	48.16	59.85
	ORG	50.58	21.29	29.96	45.08	37.68	41.05
	PER	69.53	62.67	65.92	46.53	53.70	49.86
	Overall	61.48	47.44	53.55	61.72	46.78	53.22
OpenNLP	LOC	76.54	52.22	62.08	84.34	45.84	59.40
	ORG	38.06	14.87	21.39	59.27	30.64	40.39
	PER	83.94	37.17	51.52	62.34	41.98	50.17
	Overall	68.68	30.44	42.18	37.35	41.71	39.41
SpaCy	LOC	73.38	75.36	74.36	77.04	56.64	65.28
	ORG	40.95	36.24	38.45	41.20	36.50	38.70
	PER	66.89	56.22	61.09	67.41	69.14	68.27
	Overall	60.94	49.01	54.33	66.15	54.32	59.66

TABLE III

DETAILED RESULTS OF THE EVALUATION STUDY. STANFORDNLP OUTPERFORMS THE OTHER SOFTWARE ON BOTH CORPUS, BUT THE DISPARITY IS NOT AS HIGH ON GMB THAN IT IS ON CoNLL 2003 CORPUS.

		Current study (Overall)			
		[16]	[20]	[17]	
StanfordNLP	P	70	N/A	88	90.06
	R	70	N/A	87	73.67
	F1	70	89	87	81.05
NLTK	P	88	N/A	N/A	51.78
	R	89	N/A	N/A	45.56
	F1	89	N/A	N/A	48.47
Gate	P	N/A	78	N/A	61.48
	R	N/A	74	N/A	47.44
	F1	N/A	77	N/A	53.55
OpenNLP	P	88	N/A	N/A	68.68
	R	88	N/A	N/A	30.44
	F1	88	N/A	N/A	42.18

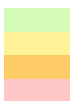

 Our result is [0; 20]% distant from the corresponding study
 Our result is [20; 40]% distant from the corresponding study
 Our result is [40; 60]% distant from the corresponding study
 Our result is [60; 100]% distant from the corresponding study

TABLE IV

RESULT COMPARISON ON CoNLL 2003 BETWEEN OUR RESULTS AND EXISTING STUDIES.

This is why we believe that our study, which aims to be as much complete and transparent as possible, should contribute to lay the groundwork for enabling NLP practitioners to take advantage of our replicable comparison study as benchmarking in future research.

V. CONCLUSION

Named Entity Recognition (NER) plays a key role in the detection and classification of entities in NLP applications. In this paper, we have shown that, despite the availability of NER

software, it remains difficult for NLP practitioners to clearly and objectively identify what software perform(s) the best. A reason for this is that most of the existing studies lack of transparency to allow for the reproducibility of experiments, adding that distinct evaluation studies – *evaluating a same software, based on a same corpus* – often lead to different results, sometimes in a substantial way.

To overcome this, we propose a replicable and comprehensive study (publicly available) to evaluate and compare 5 NER software (StanfordNLP, NLTK, OpenNLP, SpaCy, Gate) based on two distinct corpora (CoNLL 2003 and GMB). Results show that StanfordNLP outperforms the other software, in a significant manner regarding CoNLL 2003 ($\approx 30\%$ more performant, which was expected as the default classifier was trained on that corpus) and in less significant way regarding GMB ($\approx 15\%$, StanfordNLP being even less performant than NLTK at tagging "Location") for which none of the evaluated software were trained on.

Based on this finding, we more thoroughly studied different types of classifiers with StanfordNLP on both corpora, namely (i) *CoNLL 4 class*; (ii) *All 3 class* and (iii) MUC classifiers. Results show that classifiers trained on CoNLL 2003 (namely *CoNLL 4 class* and *All 3 class*) outperform MUC ($\approx 30\%$ more performant), while they provide similar results when evaluated on GMB ($\approx 5\%$ more performant). Interestingly, the MUC classifier gets better results on GMB. Most of our results are quite different from what we found in the literature, which has motivated us to design a clear and reproducible methodology. In future research, it would be worth investigating whether it would be preferable to build and train its own classifier (which may turn to be time-consuming) rather than using existing ones, even after having evaluated, compared and selected the best classifier among a set of candidates.

REFERENCES

- [1] J. Cowie and W. Lehnert. 1996. Information extraction. *Communications of the ACM* 39(1):80–91.
- [2] D. Nadeau and S. Sekine. 2007. A survey of named entity recognition and classification. *Linguisticae Investigationes* 30(1):3–26.
- [3] A. Ben Abacha and P. Zweigenbaum. 2011. Medical entity recognition: A comparison of semantic and statistical methods. In: *BioNLP Workshop. Association for Computational Linguistics. Portland, USA, 23-24 June*, pages 56–64.
- [4] I. Chalkidis, I. Androutsopoulos, and A. Michos. 2017. Extracting contract elements. In *16th International Conference on Artificial Intelligence and Law. London, UK, 12 June*, pages 19–28.
- [5] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In: *52nd Annual Meeting of the Association for Computational Linguistics. Baltimore, USA, 22-27 June*, pages 55–60.
- [6] J. Rose Finkel, T. Grenager, and C. Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In: *43rd annual meeting on association for computational linguistics. Association for Computational Linguistics, Ann Arbor, USA, 25-30 June*, pages 363–370.
- [7] A. Stent and J. D. Choi. 2015. It depends: Dependency parser comparison using a web-based evaluation tool. In: *53rd Annual Meeting of the Association for Computational Linguistics, Beijing, China, 6-11 July*, pages 13–24.
- [8] S. Bird and E. Loper. 2004. NLTK: The Natural Language Toolkit. In: *ACL 2004. Association for Computational Linguistics, Barcelona, Spain, 21-26 July*, pages 31.
- [9] E. F. Tjong Kim Sang and F. De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In: *7th Conference on Natural Language Learning at HLT-NAACL 2003-Volume 4, Edmonton, Canada, 31 May*, pages 142–147.
- [10] R. Grishman and B. Sundheim. 1996. Message Understanding Conference-6: A Brief History. In: *16th Conference on Computational Linguistics - Vol 1, Copenhagen, Denmark, 5-9 August*, pages 466–471.
- [11] N. Chinchor and P. Robinson. 1998. MUC-7 Named Entity Task Definition. In: *7th Conference on Message Understanding. Held in Fairfax, Virginia, April 29 - May 1, 1998*.
- [12] G. Doddington, A. Mitchell, M. Przybocki, L. Ramshaw, S. Strassel, and R. Weischedel. 2004. The Automatic Content Extraction (ACE) Program-Tasks, Data, and Evaluation. In: *4th International Conference on Language Resources and Evaluation. Lisbon, Portugal, 26-28 May*.
- [13] D. Balasuriya, N. Ringland, J. Nothman, T. Murphy, and J. R. Curran. 2009. Named entity recognition in Wikipedia. In: *Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources. Association for Computational Linguistics, Suntec, Singapore, 7 August*, pages 10–18.
- [14] A. C. Basave, A. Varga, M. Rowe, M. Stankovic, and A. Dadzie. 2013. In: *Making sense of microposts (#MSM2013) concept extraction challenge*. In Concept Extraction Challenge at the Workshop on 'Making Sense of Microposts', volume 1019.
- [15] T. Finin, W. Murnane, A. Karandikar, N. Keller, J. Martineau, and M. Dredze. 2010. Annotating named entities in Twitter data with crowdsourcing. In: *Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk. Association for Computational Linguistics, Los Angeles, USA, 6 June*, pages 80–88.
- [16] A. Pinto, H. Gonalo Oliveira, and A. Oliveira Alves. 2016. Comparing the performance of different NLP toolkits in formal and social media text. In: *5th Symposium on Languages, Applications and Technologies, Dagstuhl, Germany, 20-21 June*, pages 1–16.
- [17] L. Derczynski, D. Maynard, G. Rizzo, M. van Erp, G. Gorrell, R. Troncy, J. Petrak, and K. Bontcheva. 2015. Analysis of named entity recognition and linking for tweets. *Information Processing & Management* 51(2):32–49.
- [18] V. Basile, J. Bos, K. Evang et al. 2012. Developing a large semantically annotated corpus. In: *8th International Conference on Language Resources and Evaluation, Istanbul, Turkey, 21-27 May*.
- [19] E. Tjong Kim Sang. 2000. Text chunking by system combination. In: *2nd workshop on Learning language in logic. Association for Computational Linguistics, Lisbon, Portugal, 13-14 September*, pages 151–153.
- [20] K. Bontcheva, L. Derczynski, A. Funk, M. Greenwood, D. Maynard, and N. Aswani. 2013. TwitIE: An open-source information extraction pipeline for microblog text. In: *International Conference Recent Advances in Natural Language Processing. Hissar, Bulgaria, 9-11 September*, pages 83–90.
- [21] R. Jiang, R. E. Banchs, and H. Li. 2016. Evaluating and Combining Name Entity Recognition Systems. In: *6th Named Entity Workshop. Association for Computational Linguistics, Berlin, Germany, 12 August*, pages 21–27.
- [22] M. Honnibal and I. Montani. SpaCy benchmarks. <https://spacy.io/usage/facts-figures#benchmarks>.
- [23] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky. StanfordNLP NER results. <https://nlp.stanford.edu/projects/project-ner.shtml>.