Evaluation of Stanford NER for Extraction of Assembly Information from Instruction Manuals

Carlos M. Costa, Germano Veiga, Armando Sousa, Sérgio Nunes
INESC TEC and Faculty of Engineering, University of Porto, Portugal
Email: {carlos.m.costa, germano.veiga}@inesctec.pt, {asousa, sergio.nunes}@fe.up.pt

***Abstract*—Teaching industrial robots by demonstration can significantly decrease the repurposing costs of assembly lines worldwide. To achieve this goal, the robot needs to detect and track each component with high accuracy. To speedup the initial object recognition phase, the learning system can gather information from assembly manuals in order to identify which parts and tools are required for assembling a new product (avoiding exhaustive search in a large model database) and if possible also extract the assembly order and spatial relation between them. This paper presents a detailed analysis of the fine tuning of the Stanford Named Entity Recognizer for this text tagging task. Starting from the recommended configuration, it was performed 91 tests targeting the main features / parameters. Each test only changed a single parameter in relation to the recommend configuration, and its goal was to see the impact of the new configuration in the precision, recall and F1 metrics. This analysis allowed to fine tune the Stanford NER system, achieving a precision of 89.91%, recall of 83.51% and F1 of 84.69%. These results were retrieved with our new manually annotated dataset containing text with assembly operations for alternators, gearboxes and engines, which were written in a language discourse that ranges from professional to informal. The dataset can also be used to evaluate other information extraction and computer vision systems, since most assembly operations have pictures and diagrams showing the necessary product parts, their assembly order and relative spatial disposition.**

***Index Terms*—Named Entity Recognition, Natural Language Processing, Small Parts Assembly, Stanford NER**

I. Introduction

Programming of industrial robots for assembly operations is a meticulous and arduous task that requires a significant engineering effort with long testing and deployment phases. For high volume manufacturing this cost is acceptable, but it is too expensive to repurpose robots for small volume production using traditional programming approaches. These issues can be overcome with robots that can learn new assembly skills by observing experienced operators and interacting with them through natural language. To achieve these goals, the robot needs to successfully recognize the objects within its workspace and semantically track their pose with high precision while the operator demonstrates how to perform the assembly operations. Moreover, it must be able to understand any instructions that the operator might give and also have the ability to recall them if asked later on. This type of teaching allows rapid reprogramming of flexible robotic assembly cells for new tasks, but it can be speed up even further if there are assembly manuals available, which allows the robotic system to extract the objects and their assembly spatial disposition from the textual and visual representations. By knowing which objects to expect for a given teaching session, the object recognition system efficiency can be significantly increased (by limiting the object search database). Moreover, this preliminary learning phase reduces the human teaching to only the operations that lack detailed information. This type of information extraction problem is know in the Natural Language Processing (NLP) domain as Named Entity Recognition (NER), and is usually tackled with Machine Learning (ML) approaches that rely on statistical models such as Conditional Random Fields (CRFs) or Hidden Markov Models (HMMs), coupled with fine tuned regular expression matching systems and gazetteers. One of the most used NER implementation for this task is the Stanford NER[1] [1], which is integrated into the well known Stanford CoreNLP toolkit [2].

This paper provides a detailed analysis of the impact that each of the main features available in the Stanford NER system have in the overall entity recognition performance, allowing to fine tune the language model training to a given corpus. The proper selection of the CRF training features for a given application domain can have a significant effect on the entity recognition performance [3]. For example, simple token level training of CRFs leads to poor performance, but if text features such as word prefix / suffix / shape, orthographic / morphological clues, along with word / phrasal clustering and Part-Of-Speech (POS) tags are used, this can result in a CRF language model that can be very effective for recognizing the intended entities. Moreover, the performance can be improved even further if gazetteers and external knowledge databases are used.

It was performed 91 tests that started with the recommended configuration and then either changed a single parameter or enabled / disabled a given feature. This allowed to identify which features should be used in order to obtain the optimal model training configuration. Although these tests were performed with our target corpus, we expect that the features / parameters which either significantly improved or degraded the recognition performance will be transversal to the corpus used. In our new annotated dataset of assembly operations this analysis allowed to fine tune the Stanford NER system, which managed to achieve a precision of 89.91%, recall of 83.51% and F1 of 84.69%, corresponding to an improvement

978-1-5090-6234-8/17/$31.00

[1] https://nlp.stanford.edu/software/CRF-NER.shtml

of 3.23% in F1, 5.79% in recall and 0.35% in precision over the recommended configuration given in the official documentation. Our annotated dataset contained assembly instructions of alternators, gearboxes and engines in several writing styles, from highly professional and structured text to colloquial and informal language. These assembly operations were extracted from Portable Document Format (PDF) files that besides textual descriptions also had assembly pictures and diagrams. As such, this dataset can be used for evaluating systems that combine both natural language processing algorithms and also computer vision and information extraction systems. Besides token level manual annotation, each assembly operation has a list with the required parts for successfully performing the product assembly. For speeding up testing, the dataset is already split into 84% of training text and 16% of testing text.

In the following section it will be given a brief overview of applications of NLP in robotics and also the main related work on extraction of assembly information from textual representations. Section III describes the main dataset sources for the 3 product types with assembly operations. Section IV presents the main steps that were performed to extract and clean the text from the PDFs. Section V gives a brief analysis of the fine tuning results, informing what were the features / parameters that either significantly increased or decreased the recognition performance. Finally, Section VI presents the conclusions and Section VII gives an overview of possible future work.

II. Related work

NLP algorithms have been integrated into robotics systems for a myriad of applications, ranging from control of industrial robotic arms [4], [5] and mobile robots [6] to complex interaction with humanoid systems using a combination of voice, text and image perception analysis [7]–[9]. For the voice and textual teaching, the objects names and relations can be identified using NER algorithms [10], [11]. This type of approach usually relies on syntactic and semantic parsing of the text and also in machine learning algorithms [12] (such as Support Vector Machines (SVMs), HMMs and CRFs) in order to be able to recognize previously unseen object names. It may present some challenges [13], but this methodology can achieve multilingual entity recognition [14] if language agnostic attributes are used. Other complementary techniques, such as gazetteers, can improve overall recognition by providing a list of known entities that can be used for exact / partial string matching. This might be a suitable approach when we have extensive and representative entity lists and we are not expecting the text we are going to annotate to have significant new entities. Otherwise, the gazetteers might increase the number of false negatives for unseen entities. This problem might be overcome [15] by normalizing the gazetteer features and using two CRF models for text tagging (one trained with the gazetteer features and another without), that are later on combined with a logarithmic opinion pool. This is similar to a mixture model, but uses a weighted multiplicative combination of models instead of a weighted additive combination.

Advanced applications of NLP algorithms include the teaching of assembly operations to robot arm manipulation systems by human operators. The JAST robot [16] was implemented using a multi-agent system capable of learning assembly operations by interpreting human voice commands along with their gestures and gaze. The speech recognition system used a Combinatory Categorial Grammar (CCG) and a semantics module to analyze if the operator was making statements for teaching, asking for information or giving answers to previous questions made by the robot. The vision system besides tracking the hands and gaze of the operator to perform a better speech analysis, it also recognized the assembly objects within the robot workspace using template matching techniques.

Industrial applications of NLP [17] can also rely on multilingual statistical semantic parsers to extract assembly operations from natural language sentences given by remote operators. The system developed in the ROSETTA[2] research project used a client-server architecture containing a natural language parser, a Knowledge Integration Framework (KIF) and an engineering system. The parser main goal was to find predicates with their respective arguments in sentences and establish coreference chains. The KIF contained ontologies and semantically annotated skills that were used for filtering the predicates and return only the ones relevant for assembly operations. Lastly, the engineering system was a high level programming interface that used the predicates found by the parser to select the appropriate skills for assembly while also matching the predicates arguments with the knowledge database in order to identify the objects and analyze in which branch of the assembly tree they should be inserted for achieving proper assembly order.

Besides voice and textual input from humans, the assembly information can also be retrieved from online web pages or knowledge repositories. During the RoboHow[3] research project it was developed a system [18] that could extract the assembly graph by performing a syntactic and semantic analysis using a Probabilistic Context-Free Grammar (PCFG) parser and a POS tagger followed by word sense retrieval and disambiguation using the WordNet database and the Cyc ontology. After having a preliminary assembly plan, it was executed in simulation in order to perform a high level validation and allow the optimization of the robot movements. If ambiguous or missing information was detected, the system tried to generate a valid assembly plan by analyzing the objects' environment, assembly context and also similar operations stored in its knowledge database.

Named entity recognition can also be useful to identify key information from mission operation orders given to operators of robotics systems, such as Unmanned Aerial Vehicles (UAVs). Highlighting entities such as persons, times, locations, coordinates, targets and organizations allows the human operators to extract the necessary mission information faster. Moreover, in the future it might even be possible to

[2]http://www.fp7rosetta.org

[3]http://www.robohow.eu

Table I
DATASET SOURCES OVERVIEW

	Alternators	*Engines*	*Gearboxes*	*Global*
Nº of pages	84	148	221	453
Nº of assembly procedures	2	40	53	95
Nº of words	9312	22747	31798	63857
Nº of characters	58418	136297	201438	396153

have the robotic system autonomously extract all the required information to carry on the mission without human assistance. This task can be performed using a machine learning approach [19] that relies on CRF statistical models that use features such as word lists, regular expressions, prefixes / suffixes, word case and also unigram / bigram / trigrams models. The evaluation of the NER system was performed using metrics such as precision, recall and accuracy and used 9-fold cross validation for having a rotating train / test dataset in order to avoid model over-fitting.

Several NER systems and datasets have been presented over the years for news and tweets [20], [21]. This paper aims to provide an evaluation of the Stanford NER system within the assembly operations domain using our new manually annotated corpus containing a diverse range of assembly manuals for small complex objects (alternators, gearboxes and engines) written in a language discourse that ranges from professional to informal.

III. DATASET SOURCES

Our new dataset is composed of 10 English instruction manuals with 453 pages detailing assembly operations of alternators, engines and gearboxes (more details shown in Table I). These object categories were selected because they have small, light and diverse components that a typical industrial robot arm can manipulate and also because they have increasing complexity (from the simple gearboxes to the much more complex fuel / steam engines). These manuals were selected for performing NER because they are a representative sample of the several types of manuals that are available for operators working in small parts assembly and also because they were written with a language discourse ranging from very concise and professional to a more colloquial and unstructured type. Moreover they provide tables / lists with the parts and tools required for the assembly operations which are very useful for evaluating NER systems.

Most of these assembly instruction manuals are single column (two of them are dual column) and have Computer Aided Design (CAD) drawings or pictures alongside the assembly procedures. Moreover, some of these procedures are very long, with the description of all the necessary parts for the entire assembly operation while others have the assembly operations split across the main object components.

A. Alternators

Alternators are electrical generators that can convert mechanical energy into electrical energy in the form of alternating current. Their assembly is quite complex, involving a lot of small parts and intricate wire bending.

This part of the dataset includes the detailed assembly of two automotive alternators (used to power the electric equipment of cars and charge their battery). One of them was written in a dual column layout with a lot of diagrams and in a professional and concise language style while the other one was written in single column informal language discourse while using mostly pictures instead of technical diagrams.

B. Gearboxes

Gearboxes are mechanical transmission systems that provide speed and torque conversion while also giving the option of forward and backwards wheel movement. They allow a typical car engine that operates at [600, 7000] Rotations Per Minute (RPM) to move the wheels that usually rotate at [0, 1800] RPM. They can provide more torque when using lower gears and greater speed when employing higher gears. They also give the user more control over the engine performance, allowing better fuel efficiency while also reducing engine wear. Given the high variability in gearbox designs and their interconnecting gears, they can have a complex assembly sequence using mostly medium size parts.

This part of the dataset contains a detailed instruction manual for a car gearbox and another with an extensive collection of small assembly procedures for 52 industrial gearboxes (mainly used in agricultural vehicles such as tractors). Both manuals were written with a professional discourse and in a single column layout. The first had a lot of pictures and CAD drawings, while the second only had technical diagrams for each gearbox assembly procedure.

C. Engines

An engine is a machine designed to convert a given source of energy (such as fuel, electricity, compressed air, elastic / chemical energy, etc) into useful mechanical energy.

This part of the dataset provides an instruction manual with the detailed assembly procedures (35) of a small aircraft engine and also 5 more manuals with the assembly operations of small steam engines. All engine assembly manuals were written with a professional language style and had a single column structure with a lot of accompanying figures.

IV. DATASET PREPARATION

Automatic extraction of text from PDF files with multi-column text, tables and large number of images and diagrams is a challenging task for any NLP system. As such, the dataset preparation included the automatic extraction of text from the PDF files, followed by a manual cleaning and inspection phase in which all the text that was not related to assembly operations was removed. To speedup this process and ensure proper text cleaning across the entire dataset, it was applied a set of regular expressions in order to remove page headers and footers and correct formating issues related with the text extraction. After this preprocessing stage (which may not be required in deployed NLP systems, since the language models

Table II
ANNOTATED DATASET OVERVIEW

	Alternators	*Engines*	*Gearboxes*	*Global*
Nº of train tokens	4450	9101	6819	20370
Nº of test tokens	781	1852	1344	3977
Nº of PART train tokens	738	1709	1477	3924
Nº of PART test tokens	156	372	327	855
Nº of RPOS train tokens	83	258	546	887
Nº of RPOS test tokens	25	52	96	173
Nº of TOOL train tokens	72	33	33	138
Nº of TOOL test tokens	5	11	0	16
Nº of OPER train tokens	182	342	435	959
Nº of OPER test tokens	30	73	72	175
Nº of ID train tokens	2	49	76	127
Nº of ID test tokens	0	35	138	173
Nº of QTY train tokens	45	41	70	156
Nº of QTY test tokens	4	22	23	49
Nº of DIM train tokens	0	67	0	67
Nº of DIM test tokens	0	5	0	5
Nº of WGT train tokens	0	2	0	2
Nº of WGT test tokens	0	0	0	0
Nº of PROP train tokens	36	63	2	101
Nº of PROP test tokens	13	2	0	15

Table III
ALTERNATORS DATASET ANNOTATIONS

Token	*Class*
Use	O
5/16	DIM
"	DIM
hex	TOOL
wrench	TOOL
or	O
5/16	DIM
"	DIM
hex	TOOL
drive	TOOL
,	O
in	O
the	O
end	RPOS
of	RPOS
the	O
shaft	PART
,	O
to	O
hold	OPER
while	O
removing	OPER
shaft	PART
nut	PART
.	O

Table IV
ENGINES DATASET ANNOTATIONS

Token	*Class*
Remove	OPER
the	O
black	PROP
outer	RPOS
jacket	PART
and	O
the	O
white	PROP
insulation	PART
core	PART
to	O
expose	OPER
a	O
3/8	DIM
"	DIM
length	DIM
of	O
the	O
inner	RPOS
conductor	PART
.	O

Table V
GEARBOXES DATASET ANNOTATIONS

Token	*Class*
Install	OPER
Slotted	PART
Hex	PART
Bearing	PART
Adjusting	PART
Nut	PART
(	O
#	ID
770730	ID
)	O
with	O
Cotter	TOOL
Pin	TOOL
(	O
#	ID
770421	ID
)	O
.	O

are already computed), the dataset was proofread to correct spelling errors. Later on the lists / tables with the information about the required assembly parts / tools was moved into validation files in order to allow the evaluation of information extraction systems.

Given that the evaluation of NER systems normally requires text annotated with the expected entities at the token level, a small yet representative part of the dataset was manually annotated using the IO (Inside / Outside) encoding and saved in the Tab Separated Value (TSV) file format expected by the Stanford NER system (examples of annotations in Tables III to V). This subset of he dataset contained assembly operations from the 3 categories of objects (alternators, engines, gearboxes) and was annotated with 9 types of entities (detailed entity counts in Table II) plus a neutral class that represents no entity (O). Most of them are product parts (PART), their relative position (RPOS) and the operations (OPER) in which they are involved, followed by the tools (TOOL) required. Other minority classes include the parts unique identification numbers (ID), their quantity (QTY), dimensions (DIM), weight (WGT) and general properties (PROP), such as surface color.

In order to speeding up testing for other developers, the dataset is already split into 84% of training text and 16% of test text and is available at[4].

V. STANFORD NER TUNING ANALYSIS

Analyzing Tables VI and VII it can be seen that the fine tuning of the CRF training features allowed to achieve an absolute improvement (in relation to the official recommended configuration) of 3.23% in F1, 5.79% in recall and 0.35% in precision, resulting in an overall entity recognition performance of 85.91% in precision, 83.51% in recall and 84.69% in F1. As expected, the entity classes with more training samples (such as PART, OPER, RPOS) performed much better than the classes with few instances (which were the case of TOOL, PROP, DIM, QTY, WGT), with the exception of the ID class, that although it had a modest number of samples, it was the best performing entity class, with a F1 metric of 97.87%. This was due to the effectiveness of the word shape and previous word features that were able to capture the fact that in our corpus the IDs usually contain the "#" or "ACV" prefix or are preceded with "P/N".

Looking at Table VIII it can be seen that in isolation most of the main configuration parameters / features available in the Stanford NER do not affect the F1 metric significantly. The feature that provided the best improvement was the lowercasing of the words for the n-gram language models, which provided a relative boost of 1.6%. This is due to the fact that in some assembly operations the entities are capitalized while in other they are not. As such, lowercasing helps to avoid this issue in the testing corpus.

Other features that also improved the F1 metric were the adjusting of the word shape algorithm parameters, along with the increase of the n-gram model length and usage of the middle n-grams. As expected, the usage of gazetteers (built from the manuals list of necessary parts and tools along with

[4]https://github.com/carlosmccosta/Assembly-Named-Entity-Recognition

the provided IDs) also helped to slightly improve the recognition performance. Besides tuning and activating processing modules, disabling the class features and the usage of the next word also helped improved the F1 metric. Changing the inference type algorithm from Viterbi to Beam slightly improved the recognition performance while increasing the CRF order from 1 to 2 significantly increase the computation time (10 times more) with almost no improvement in recognition performance.

On the other hand, looking at Tables IX and X it can be seen that disabling or lowering the n-gram model length to just unigrams was the change that most decreased the F1 metric (with a relative reduction of 8.2%), followed closely by the usage of the bag of words features and the sigma smoothing. Disabling the previous word feature also decreased the recognition performance (with a relative decrease of F1 metric of 1.2%).

When inspecting the training time, it can be seen that disabling the usage of sequences and previous sequences, along with disabling the usage of the previous word and activating the usage of only the observed sequences managed to reduce the training time to almost half. On the other hand, using the bag of words features drastically increased the training time (up to 50 times higher) but it managed to reduce the false positives by 22.1% (while almost tripling the false positives).

Moving to the tagging time performance, we can see that the Viterbi inference algorithm is almost twice as fast as the Beam algorithm while being able to achieve the same recognition performance. Moreover, increasing the CRF order from 1 to 2 or using the bag of words features made the tagger run around 5 times slower.

Monitoring the memory usage we can confirm that reducing the number of past guesses used by the limited memory quasi Newton optimizer (L-BFGS) from 25 to 2 managed to reduce the maximum memory consumption from 2.7 GB to 1.8 GB with very low impact on the recognition performance (less than 1%). We can also confirm that dropping features with a absolute weight value below 0.1 allowed to reduce the serialized model size from 16.5 MB to 7.6 MB while improving the CRF tagging speed by 10% and slightly increasing the F1 metric (less than 1%).

There are a lot of more parameters and features than the ones previously discussed and presented in Tables VIII to X that can be fine tuned. Moreover, given how easy it is to add new ones to the Stanford NER processing pipeline and the growing community using the CoreNLP toolkit, we think that this analysis is a useful starting point to anyone interested in using and improving the Stanford NER tagging system.

VI. Conclusions

This paper presented a detailed analysis of the fine tuning of the Stanford NER system in our annotated corpus with assembly operations. It was performed 91 tests targeting the main configurations / features of the Stanford NER implementation, in which the recommended configuration was used as a starting point and then each test either changed a numeric parameter or enabled / disabled a given feature. This allowed to select the best training configuration for the statistical language models (CRFs), which achieved 85.91% in precision, 83.51% in recall and 84.69% in F1 (corresponding to an improvement of 3.23% in F1, 5.79% in recall and 0.35% in precision over the recommended configuration given in the official documentation). Although these tests were performed with our target corpus, we expect that the parameters / features which either significantly improved or degraded the NER recognition performance will be transversal to the dataset used, allowing other researchers to use this configuration as a starting point for their specific corpus.

Our dataset contains assembly operations of alternators, engines and gearboxes in textual form and is complemented with object pictures and assembly diagrams. For evaluating NER systems using this dataset, each assembly operation has associated a list with the parts and quantities needed to successfully perform the product assembly. In order to have a representative dataset, it is provided assembly operations written in a professional / structured manner and also in an informal and colloquial language register. Moreover, a small yet representative part is manually annotated at the token level in a TSV format.

This dataset was built for evaluating NER systems, but can also be used to evaluate information extraction and computer vision systems, given the large textual and image information that it provides for each assembly operation.

Table VI
NER results using the recommended configuration

Entity	*Precision*	*Recall*	*F1*	*True positives*	*False positives*	*False negatives*
DIM	0.0714	0.5000	0.1250	1	13	1
ID	0.9886	0.9158	0.9508	87	1	8
OPER	0.8896	0.8286	0.8580	145	18	30
PART	0.8513	0.7740	0.8108	435	76	127
PROP	0.8889	0.5333	0.6667	8	1	7
QTY	0.3333	0.1724	0.2273	5	10	24
RPOS	0.8777	0.8188	0.8472	122	17	27
TOOL	1.0000	0.3000	0.4615	3	0	7
Totals	**0.8556**	**0.7772**	**0.8146**	**806**	**136**	**231**

Table VII
NER results using the fine tuned configuration

Entity	*Precision*	*Recall*	*F1*	*True positives*	*False positives*	*False negatives*
DIM	0.0625	0.5000	0.1111	1	15	1
ID	0.9892	0.9684	0.9787	92	1	3
OPER	0.8935	0.8629	0.8779	151	18	24
PART	0.8667	0.8327	0.8494	468	72	94
PROP	0.6000	0.6000	0.6000	9	6	6
QTY	0.5926	0.5517	0.5714	16	11	13
RPOS	0.8681	0.8389	0.8532	125	19	24
TOOL	1.0000	0.4000	0.5714	4	0	6
Totals	**0.8591**	**0.8351**	**0.8469**	**866**	**142**	**171**

VII. Future work

The main goals of this paper were the creation of a dataset for NER within the domain of assembly operations along with the testing of the effectiveness of state of the art NER systems for tagging / extraction of the intended textual entities. Future work would include the development of a system capable of build on top of a given NER processing pipeline and create the assembly graph containing in each node the installation of a given part with associated metadata, such as the tools required, its relative position in relation to already assembled parts and other information that may be useful to other higher level systems (for example the part ID, its dimensions and weight along with other related information, such as the torque necessary to apply when performing mating / screwing operations). Moreover, it would be useful to manually annotate the remaining of the dataset and add other types of assembly operations in order to broaden the scope of the presented corpus.

Appendix

The appendix contains Tables VIII to X with the detailed fine tuning results of the Stanford NER system when using our new corpus of assembly operations. Each line in the tables corresponds to a single test in which a given parameter / feature was changed in relation to the recommended configuration. Besides the standard NER evaluation metrics (precision, recall, F1 with associated true positives, false positives and false negatives) it is also provided the language model training time and the classifier performance (in words / second) when tagging the test corpus. Moreover, to allow easy analysis of the test results, it is provided extra columns with the relative performance in relation to the recommended configuration results (each relative result is computed by dividing the current test result with the corresponding result achieved with the recommended configuration given in the official documentation).

Acknowledgment

The research leading to these results has received funding from the European Union's Horizon 2020 - The EU Framework Programme for Research and Innovation 2014-2020, under grant agreement No. 723658.

References

[1] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ser. ACL '05. Stroudsburg, PA, USA: Association for Computational Linguistics, 2005, pp. 363–370.

[2] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2014, pp. 55–60.

[3] M. Tkachenko and A. Simanovsky, "Named entity recognition: Exploring features," in *Proceedings of KONVENS 2012*, J. Jancsary, Ed. ÖGAI, September 2012, pp. 118–127, main track: oral presentations.

[4] B. Akan, A. Ameri, B. Curuklu, and L. Asplund, "Intuitive industrial robot programming through incremental multimodal language and augmented reality," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, May 2011, pp. 3934–3939.

[5] K. Watanabe, C. Jayawardena, and K. Izumi, "Approximate decision making by natural language commands for robots," in *32nd IEEE Annual Conference on Industrial Electronics*, November 2006, pp. 4480–4485.

[6] C. Matuszek, E. Herbst, L. Zettlemoyer, and D. Fox, *Experimental Robotics: The 13th International Symposium on Experimental Robotics*. Springer International Publishing, 2013, ch. Learning to Parse Natural Language Commands to a Robot Control System, pp. 403–415.

[7] E. Neo, T. Sakaguchi, and K. Yokoi, "A humanoid robot that listens, speaks, sees and manipulates in human environments," in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, August 2008, pp. 419–425.

[8] P. Barabas, L. Kovacs, and M. Vircikova, "Robot controlling in natural language," in *IEEE 3rd International Conference on Cognitive Infocommunications (CogInfoCom)*, December 2012, pp. 181–186.

[9] A. Antunes, L. Jamone, G. Saponaro, A. Bernardino, and R. Ventura, "From human instructions to robot actions: Formulation of goals, affordances and probabilistic planning," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 5449–5454.

[10] L. Derczynski, D. Maynard, G. Rizzo, M. van Erp, G. Gorrell, R. Troncy, J. Petrak, and K. Bontcheva, "Analysis of named entity recognition and linking for tweets," *CoRR*, vol. abs/1410.7182, 2014.

[11] S. Dlugolinsky, M. Ciglan, and M. Laclavik, "Evaluation of named entity recognition tools on microposts," in *IEEE 17th International Conference on Intelligent Engineering Systems (INES)*, June 2013, pp. 197–202.

[12] A. Ekbal, S. Saha, and D. Singh, "Active machine learning technique for named entity recognition," in *Proceedings of the International Conference on Advances in Computing, Communications and Informatics*. ACM, 2012, pp. 180–186.

[13] L. Ratinov and D. Roth, "Design challenges and misconceptions in named entity recognition," in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, ser. CoNLL '09. Association for Computational Linguistics, 2009, pp. 147–155.

[14] R. Al-Rfou, V. Kulkarni, B. Perozzi, and S. Skiena, "Polyglot-NER: Massive multilingual named entity recognition," *Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, British Columbia, Canada, April 30 - May 2, 2015*, April 2015.

[15] A. Smith and M. Osborne, "Using gazetteers in discriminative information extraction," in *Proceedings of the Tenth Conference on Computational Natural Language Learning*, ser. CoNLL-X '06. Stroudsburg, PA, USA: Association for Computational Linguistics, 2006, pp. 133–140.

[16] M. Rickert, M. E. Foster, M. Giuliani, T. By, G. Panin, and A. Knoll, "Integrating language, vision and action for human robot dialog systems," in *Proceedings of the International Conference on Universal Access in Human-Computer Interaction, HCI International*, ser. Lecture Notes in Computer Science, vol. 4555. Beijing, China: Springer, 2007, pp. 987–995.

[17] M. Stenmark and P. Nugues, "Natural language programming of industrial robots," in *44th International Symposium on Robotics (ISR)*, October 2013, pp. 1–5.

[18] M. Tenorth, D. Nyga, and M. Beetz, "Understanding and executing instructions for everyday manipulation tasks from the world wide web," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2010, pp. 1486–1491.

[19] D. Chesworth, N. Harmon, L. Tanner, S. Guerlain, and M. Balazs, "Named-entity recognition and data visualization techniques to communicate mission command to autonomous systems," in *2016 IEEE Systems and Information Engineering Design Symposium (SIEDS)*, April 2016, pp. 233–238.

[20] M. Dojchinovski and T. Kliegr, "Datasets and gate evaluation framework for benchmarking wikipedia-based ner systems," in *The 12th International Semantic Web Conference (ISWC2013)*, 2013.

[21] M. Röder, R. Usbeck, S. Hellmann, D. Gerber, and A. Both, "N3 - a collection of datasets for named entity recognition and disambiguation in the nlp interchange format," in *The 9th edition of the Language Resources and Evaluation Conference*, 2014.

Table VIII
TUNING OF THE NER LANGUAGE MODEL - TRAINING PARAMETERS THAT IMPROVED THE F1 METRIC

Changed parameter	*Recommended value*	*Test value*	*Train time (sec)*	*Classifier test speed (words / sec)*	*Precision*	*Recall*	*F1*	*True positives*	*False positives*	*False negatives*	*Relative train time (sec)*	*Relative classifier test speed (words / sec)*	*Relative precision*	*Relative recall*	*Relative F1*	*Relative true positives*	*Relative false positives*	*Relative false negatives*
recommended-configuration	-	-	125.5	3798.47	0.8556	0.7772	0.8146	806	136	231	-	-	-	-	-	-	-	-
optimal-configuration	-	-	176.4	4581.80	0.8591	0.8351	0.8469	866	142	171	1.406	1.206	1.004	1.074	1.040	1.074	1.044	0.740
lowercaseNGrams	FALSE	TRUE	142.5	3929.84	0.8662	0.7927	0.8278	822	127	215	1.135	1.035	1.012	1.020	1.016	1.020	0.934	0.931
wordShape	chris2useLC	chris2	130.0	4332.24	0.8580	0.7985	0.8272	828	137	209	1.036	1.141	1.003	1.027	1.015	1.027	1.007	0.905
useDisjunctive	TRUE	FALSE	137.2	4217.39	0.8670	0.7859	0.8245	815	125	222	1.093	1.110	1.013	1.011	1.012	1.011	0.919	0.961
wordShape	chris2useLC	dan1	161.1	4389.62	0.8589	0.7927	0.8245	822	135	215	1.284	1.156	1.004	1.020	1.012	1.020	0.993	0.931
noMidNGrams	TRUE	FALSE	136.2	3658.69	0.8549	0.7956	0.8242	825	140	212	1.085	0.963	0.999	1.024	1.012	1.024	1.029	0.918
wordFunction	null	Lowercase And Americanize Function	138.3	3980.98	0.8515	0.7965	0.8231	826	144	211	1.102	1.048	0.995	1.025	1.010	1.025	1.059	0.913
useClassFeature	TRUE	FALSE	125.0	3455.26	0.8617	0.7869	0.8226	816	131	221	0.996	0.910	1.007	1.012	1.010	1.012	0.963	0.957
wordShape	chris2useLC	chris1	159.6	4013.12	0.8568	0.7907	0.8225	820	137	217	1.272	1.057	1.001	1.017	1.010	1.017	1.007	0.939
useTypeSeqs	TRUE	FALSE	118.9	4121.24	0.8597	0.7859	0.8212	815	133	222	0.947	1.085	1.005	1.011	1.008	1.011	0.978	0.961
wordShape	chris2useLC	chris3	118.5	4230.85	0.8509	0.7927	0.8208	822	144	215	0.944	1.114	0.995	1.020	1.008	1.020	1.059	0.931
maxNGramLeng	6	7	128.6	4058.16	0.8603	0.7840	0.8204	813	132	224	1.025	1.068	1.005	1.009	1.007	1.009	0.971	0.970
useNext	TRUE	FALSE	115.5	4029.38	0.8611	0.7830	0.8202	812	131	225	0.920	1.061	1.006	1.007	1.007	1.007	0.963	0.974
useObservedSequencesOnly	FALSE	TRUE	65.1	5418.26	0.8586	0.7850	0.8202	814	134	223	0.519	1.426	1.004	1.010	1.007	1.010	0.985	0.965
QNsize	25	5	165.1	4212.92	0.8586	0.7850	0.8202	814	134	223	1.316	1.109	1.004	1.010	1.007	1.010	0.985	0.965
disjunctionWidth	4	5	134.8	4134.10	0.8643	0.7801	0.8201	809	127	228	1.074	1.088	1.010	1.004	1.007	1.004	0.934	0.987
maxNGramLeng	6	8	141.2	4262.59	0.8594	0.7840	0.8200	813	133	224	1.125	1.122	1.004	1.009	1.007	1.009	0.978	0.970
useLC	FALSE	TRUE	136.8	3961.16	0.8553	0.7869	0.8197	816	138	221	1.090	1.043	1.000	1.012	1.006	1.012	1.015	0.957
useTypeSeqs2	TRUE	FALSE	126.8	4142.71	0.8561	0.7859	0.8195	815	137	222	1.010	1.091	1.001	1.011	1.006	1.011	1.007	0.961
useTags	FALSE	TRUE	120.4	4262.59	0.8667	0.7772	0.8195	806	124	231	0.959	1.122	1.013	1.000	1.006	1.000	0.912	1.000
useEntityTypeSequences	FALSE	TRUE	142.3	4239.87	0.8617	0.7811	0.8194	810	130	227	1.134	1.116	1.007	1.005	1.006	1.005	0.956	0.983
disjunctionWidth	4	1	118.2	4208.47	0.8675	0.7763	0.8193	805	123	232	0.942	1.108	1.014	0.999	1.006	0.999	0.904	1.004
useNextSequences	FALSE	TRUE	146.9	4204.02	0.8599	0.7811	0.8186	810	132	227	1.171	1.107	1.005	1.005	1.005	1.005	0.971	0.983
featureDiffThresh	0	0.025	221.2	3010.60	0.8582	0.7821	0.8184	811	134	226	1.763	0.793	1.003	1.006	1.005	1.006	0.985	0.978
useEntityTypes	FALSE	TRUE	122.5	4204.02	0.8622	0.7782	0.8180	807	129	230	0.976	1.107	1.008	1.001	1.004	1.001	0.949	0.996
wordShape	chris2useLC	dan2	143.2	4100.00	0.8502	0.7878	0.8178	817	144	220	1.141	1.079	0.994	1.014	1.004	1.014	1.059	0.952
wordShape	chris2useLC	dan2bio	138.8	4271.75	0.8502	0.7878	0.8178	817	144	220	1.106	1.125	0.994	1.014	1.004	1.014	1.059	0.952
wordShape	chris2useLC	jenny1	137.8	4529.61	0.8502	0.7878	0.8178	817	144	220	1.098	1.192	0.994	1.014	1.004	1.014	1.059	0.952
dehyphenateNGrams	FALSE	TRUE	121.1	3925.96	0.8596	0.7792	0.8174	808	132	229	0.965	1.034	1.005	1.003	1.003	1.002	0.971	0.991
featureDiffThresh	0	0.05	232.4	4248.93	0.8571	0.7811	0.8174	810	135	227	1.852	1.119	1.002	1.005	1.003	1.005	0.993	0.983
useReverse	FALSE	TRUE	123.9	4017.17	0.8547	0.7830	0.8173	812	138	225	0.987	1.058	0.999	1.007	1.003	1.007	1.015	0.974
useWordPairs	FALSE	TRUE	140.8	4155.69	0.8619	0.7763	0.8168	805	129	232	1.122	1.094	1.007	0.999	1.003	0.999	0.949	1.004
useOrdinal	FALSE	TRUE	119.8	4013.12	0.8570	0.7801	0.8168	809	135	228	0.955	1.057	1.002	1.004	1.003	1.004	0.993	0.987
QNsize	25	2	232.9	4290.18	0.8627	0.7753	0.8167	804	128	233	1.856	1.129	1.008	0.998	1.003	0.998	0.941	1.009
cleanGazette	FALSE	TRUE	114.9	4341.70	0.8627	0.7753	0.8167	804	128	233	0.916	1.143	1.008	0.998	1.003	0.998	0.941	1.009
conjoinShapeNGrams	FALSE	TRUE	127.3	3914.37	0.8475	0.7878	0.8166	817	147	220	1.014	1.031	0.991	1.014	1.002	1.014	1.081	0.952
useEntityRule	FALSE	TRUE	121.9	4083.16	0.8553	0.7811	0.8165	810	137	227	0.971	1.075	1.000	1.005	1.002	1.005	1.007	0.983
QNsize	25	10	142.0	4438.62	0.8553	0.7811	0.8165	810	137	227	1.131	1.169	1.000	1.005	1.002	1.005	1.007	0.983
useLastRealWord	FALSE	TRUE	123.3	4177.52	0.8585	0.7782	0.8164	807	133	230	0.982	1.100	1.003	1.001	1.002	1.001	0.978	0.996
sloppyGazette	FALSE	TRUE	136.8	3980.98	0.8568	0.7792	0.8162	808	135	229	1.090	1.048	1.001	1.003	1.002	1.002	0.993	0.991
wordShape	chris2useLC	chris4	113.0	3914.37	0.8559	0.7792	0.8157	808	136	229	0.900	1.031	1.000	1.003	1.001	1.002	1.000	0.991
inferenceType	Viterbi	Beam	118.2	2185.16	0.8558	0.7782	0.8152	807	136	230	0.942	0.575	1.000	1.001	1.001	1.001	1.000	0.996

Table IX

TUNING OF THE NER LANGUAGE MODEL - TRAINING PARAMETERS THAT DID NOT CHANGE THE F1 METRIC

Changed parameter	*Recommended value*	*Test value*	*Train time (sec)*	*Classifier test speed (words / sec)*	*Precision*	*Recall*	*F1*	*True positives*	*False positives*	*False negatives*	*Relative train time (sec)*	*Relative classifier test speed (words / sec)*	*Relative precision*	*Relative recall*	*Relative F1*	*Relative true positives*	*Relative false positives*	*Relative false negatives*
recommended-configuration	-	-	125.5	3798.47	0.8556	0.7772	0.8146	806	136	231	-	-	-	-	-	-	-	-
optimal-configuration	-	-	176.4	4581.80	0.8591	0.8351	0.8469	866	142	171	1.406	1.206	1.004	1.074	1.040	1.074	1.044	0.740
useWord	TRUE	FALSE	137.9	3899.02	0.8614	0.7734	0.8150	802	129	235	1.099	1.026	1.007	0.995	1.000	0.995	0.949	1.017
featureDiffThresh	0	0.1	205.3	4181.91	0.8565	0.7772	0.8150	806	135	231	1.636	1.101	1.001	1.000	1.000	1.000	0.993	1.000
maxLeft	1	2	1192.2	728.39	0.8573	0.7763	0.8148	805	134	232	9.500	0.192	1.002	0.999	1.000	0.999	0.985	1.004
disjunctionWidth	4	3	115.8	4217.39	0.8630	0.7715	0.8147	800	127	237	0.923	1.110	1.009	0.993	1.000	0.993	0.934	1.026
disjunctionWidth	4	2	116.1	4258.03	0.8630	0.7715	0.8147	800	127	237	0.925	1.121	1.009	0.993	1.000	0.993	0.934	1.026
initWithNERPosterior	FALSE	TRUE	117.6	4276.34	0.8556	0.7772	0.8146	806	136	231	0.937	1.126	1.000	1.000	1.000	1.000	1.000	1.000
applyNERPenalty	TRUE	FALSE	118.4	4235.36	0.8556	0.7772	0.8146	806	136	231	0.943	1.115	1.000	1.000	1.000	1.000	1.000	1.000
useKnownLCWords	TRUE	FALSE	118.2	4346.45	0.8556	0.7772	0.8146	806	136	231	0.942	1.144	1.000	1.000	1.000	1.000	1.000	1.000
useTitle	FALSE	TRUE	118.1	4173.14	0.8556	0.7772	0.8146	806	136	231	0.941	1.099	1.000	1.000	1.000	1.000	1.000	1.000
useLongSequences	FALSE	TRUE	116.9	4049.90	0.8556	0.7772	0.8146	806	136	231	0.931	1.066	1.000	1.000	1.000	1.000	1.000	1.000
useNextRealWord	FALSE	TRUE	113.2	4155.69	0.8605	0.7734	0.8146	802	130	235	0.902	1.094	1.006	0.995	1.000	0.995	0.956	1.017
normalize	FALSE	TRUE	116.0	4493.79	0.8556	0.7772	0.8146	806	136	231	0.924	1.183	1.000	1.000	1.000	1.000	1.000	1.000
normalizeTimex	FALSE	TRUE	116.1	4262.59	0.8556	0.7772	0.8146	806	136	231	0.925	1.122	1.000	1.000	1.000	1.000	1.000	1.000
useBoundarySequences	FALSE	TRUE	116.4	4244.40	0.8556	0.7772	0.8146	806	136	231	0.927	1.117	1.000	1.000	1.000	1.000	1.000	1.000
useSymTags	FALSE	TRUE	116.6	4091.56	0.8556	0.7772	0.8146	806	136	231	0.929	1.077	1.000	1.000	1.000	1.000	1.000	1.000
useTaggySequences	FALSE	TRUE	117.1	4021.23	0.8556	0.7772	0.8146	806	136	231	0.933	1.059	1.000	1.000	1.000	1.000	1.000	1.000
useOccurrencePatterns	FALSE	TRUE	116.9	4221.87	0.8556	0.7772	0.8146	806	136	231	0.931	1.111	1.000	1.000	1.000	1.000	1.000	1.000
useViterbi	TRUE	FALSE	116.3	4322.83	0.8556	0.7772	0.8146	806	136	231	0.927	1.138	1.000	1.000	1.000	1.000	1.000	1.000
includeFullCRFInLOP	FALSE	TRUE	121.7	4199.58	0.8556	0.7772	0.8146	806	136	231	0.970	1.106	1.000	1.000	1.000	1.000	1.000	1.000
backpropLopTraining	FALSE	TRUE	122.0	3969.06	0.8556	0.7772	0.8146	806	136	231	0.972	1.045	1.000	1.000	1.000	1.000	1.000	1.000
useOutputLayer	TRUE	FALSE	121.2	4443.58	0.8556	0.7772	0.8146	806	136	231	0.966	1.170	1.000	1.000	1.000	1.000	1.000	1.000
useHiddenLayer	TRUE	FALSE	116.4	4134.10	0.8556	0.7772	0.8146	806	136	231	0.927	1.088	1.000	1.000	1.000	1.000	1.000	1.000
restrictLabels	TRUE	FALSE	112.4	4327.53	0.8556	0.7772	0.8146	806	136	231	0.896	1.139	1.000	1.000	1.000	1.000	1.000	1.000
QNsize	25	50	117.2	4160.04	0.8556	0.7772	0.8146	806	136	231	0.934	1.095	1.000	1.000	1.000	1.000	1.000	1.000
useNeighborNGrams	FALSE	TRUE	209.4	3827.72	0.8564	0.7763	0.8144	805	135	232	1.669	1.008	1.001	0.999	1.000	0.999	0.993	1.004
useTypeySequences	TRUE	FALSE	125.2	4299.46	0.8540	0.7782	0.8143	807	138	230	0.998	1.132	0.998	1.001	1.000	1.001	1.015	0.996

Table X

TUNING OF THE NER LANGUAGE MODEL - TRAINING PARAMETERS THAT DECREASED THE F1 METRIC

Changed parameter	*Recommended value*	*Test value*	*Train time (sec)*	*Classifier test speed (words / sec)*	*Precision*	*Recall*	*F1*	*True positives*	*False positives*	*False negatives*	*Relative train time (sec)*	*Relative classifier test speed (words / sec)*	*Relative precision*	*Relative recall*	*Relative F1*	*Relative true positives*	*Relative false positives*	*Relative false negatives*
recommended-configuration	-	-	125.5	3798.47	0.8556	0.7772	0.8146	806	136	231	-	-	-	-	-	-	-	-
optimal-configuration	-	-	176.4	4581.80	0.8591	0.8351	0.8469	866	142	171	1.406	1.206	1.004	1.074	1.040	1.074	1.044	0.740
use2W	FALSE	TRUE	130.8	4160.04	0.8519	0.7763	0.8123	805	140	232	1.042	1.095	0.996	0.999	0.997	0.999	1.029	1.004
wordShape	chris2useLC	chris3useLC	124.1	4248.93	0.8488	0.7743	0.8099	803	143	234	0.989	1.119	0.992	0.996	0.994	0.996	1.051	1.013
useSequences	TRUE	FALSE	64.8	3225.47	0.8402	0.7811	0.8096	810	154	227	0.516	0.849	0.982	1.005	0.994	1.005	1.132	0.983
usePrevSequences	TRUE	FALSE	65.2	4108.47	0.8402	0.7811	0.8096	810	154	227	0.520	1.082	0.982	1.005	0.994	1.005	1.132	0.983
maxNGramLeng	6	5	116.1	4498.87	0.8541	0.7676	0.8085	796	136	241	0.925	1.184	0.998	0.988	0.993	0.988	1.000	1.043
disjunctionWidth	4	6	118.7	4327.53	0.8484	0.7715	0.8081	800	143	237	0.946	1.139	0.992	0.993	0.992	0.993	1.051	1.026
wordShape	chris2useLC	dan2useLC	119.4	4313.45	0.8497	0.7686	0.8071	797	141	240	0.951	1.136	0.993	0.989	0.991	0.989	1.037	1.039
wordShape	chris2useLC	dan2bioUseLC	119.3	4394.48	0.8497	0.7686	0.8071	797	141	240	0.951	1.157	0.993	0.989	0.991	0.989	1.037	1.039
wordShape	chris2useLC	jenny1useLC	118.5	4095.78	0.8497	0.7686	0.8071	797	141	240	0.944	1.078	0.993	0.989	0.991	0.989	1.037	1.039
sigma	1	2	158.2	4235.36	0.8494	0.7666	0.8059	795	141	242	1.261	1.115	0.993	0.986	0.989	0.986	1.037	1.048
usePrev	TRUE	FALSE	65.1	3961.16	0.8328	0.7782	0.8046	807	162	230	0.519	1.043	0.973	1.001	0.988	1.001	1.191	0.996
sigma	1	3	253.8	4208.47	0.8379	0.7724	0.8038	801	155	236	2.022	1.108	0.979	0.994	0.987	0.994	1.140	1.022
useDisjunctiveShapeInteraction	FALSE	TRUE	149.7	4404.21	0.8534	0.7522	0.7996	780	134	257	1.193	1.159	0.997	0.968	0.982	0.968	0.985	1.113
maxNGramLeng	6	4	143.0	4074.80	0.8430	0.7560	0.7972	784	146	253	1.139	1.073	0.985	0.973	0.979	0.973	1.074	1.095
sigma	1	5	334.9	3751.89	0.8358	0.7608	0.7966	789	155	248	2.669	0.988	0.977	0.979	0.978	0.979	1.140	1.074
maxNGramLeng	6	3	128.9	4370.33	0.8390	0.7435	0.7883	771	148	266	1.027	1.151	0.981	0.957	0.968	0.957	1.088	1.152
useMoreNeighborNGrams	FALSE	TRUE	141.0	3933.73	0.8348	0.7454	0.7876	773	153	264	1.124	1.036	0.976	0.959	0.967	0.959	1.125	1.143
sigma	1	10	412.8	4112.72	0.8328	0.7348	0.7807	762	153	275	3.289	1.083	0.973	0.945	0.958	0.945	1.125	1.190
maxNGramLeng	6	2	132.6	4199.58	0.8247	0.7213	0.7695	748	159	289	1.057	1.106	0.964	0.928	0.945	0.928	1.169	1.251
sigma	1	20	630.6	4212.92	0.8230	0.7175	0.7666	744	160	293	5.025	1.109	0.962	0.923	0.941	0.923	1.176	1.268
sigma	1	30	758.1	3748.35	0.8145	0.7155	0.7618	742	169	295	6.041	0.987	0.952	0.921	0.935	0.921	1.243	1.277
useBagOfWords	FALSE	TRUE	6362.6	497.56	0.6951	0.8264	0.7551	857	376	180	50.698	0.131	0.812	1.063	0.927	1.063	2.765	0.779
useNGrams	TRUE	FALSE	148.9	3827.72	0.8138	0.6914	0.7477	717	164	320	1.186	1.008	0.951	0.890	0.918	0.890	1.206	1.385
maxNGramLeng	6	1	138.0	4177.52	0.8138	0.6914	0.7477	717	164	320	1.100	1.100	0.951	0.890	0.918	0.890	1.206	1.385