A Simple Model for Improving the Performance of the Stanford Parser for Action Detection in Textual Instructions

Kristina Y. Yordanova
University of Rostock
18059 Rostock
Germany
kristina.yordanova@uni-rostock.de

Abstract

Different approaches for behaviour understanding rely on textual instructions to generate models of human behaviour. These approaches usually use state of the art parsers to obtain the part of speech (POS) meaning and dependencies of the words in the instructions. For them it is essential that the parser is able to correctly annotate the instructions and especially the verbs as they describe the actions of the person. State of the art parsers usually make errors when annotating textual instructions, as they have short sentence structure often in imperative form. The inability of the parser to identify the verbs results in the inability of behaviour understanding systems to identify the relevant actions. To address this problem, we propose a simple rule-based model that attempts to correct any incorrectly annotated verbs. We argue that the model is able to significantly improve the parser's performance without the need of additional training data. We evaluate our approach by extracting the actions from 61 textual instructions annotated only with the Stanford parser and once again after applying our model. The results show a significant improvement in the recognition rate when applying the rules (75% accuracy compared to 68% without the rules, p-value $<$ 0.001).

1 Introduction and Motivation

There are various activity recognition approaches that rely on symbolic behaviour models and sensor data to reason about the user actions and goals (Yordanova and Kirste, 2015; Hiatt et al., 2011; Ramirez and Geffner, 2011). Such approaches use manually built models of the human behaviour that describe the possible actions a person can execute and the constraints associated with their execution. One problem with these approaches is that the modelling is time consuming and error-prone process (Nguyen et al., 2013; Krüger et al., 2012).

To address this problem, some works propose the automatic generation of the behaviour model from textual instructions (Yordanova, 2017b; Branavan et al., 2010). More precisely, one can utilise the knowledge encoded in textual instructions to learn the model structure. Textual instructions specify tasks for achieving a given goal without explicitly stating all the required steps. On the one hand, this makes them a challenging source for learning a model (Branavan et al., 2010). On the other hand, they are usually written in imperative form, have a simple sentence structure, and are highly organised. Compared to rich texts, this makes them a better source for identifying the sequence of actions needed for reaching the goal (Zhang et al., 2012).

To identify the set of possible actions the person can execute, these approaches rely on parsers, which assign part of speech (POS) tags to words (Yordanova, 2017b; Yordanova and Kirste, 2016; Yordanova, 2016; Preum et al., 2017; Lindsay et al., 2017). These parsers usually rely on training data in order to be able to perform POS-tagging (Martins et al., 2010; Chen and Manning, 2014). The training data comes from corpora containing texts such as news paper articles, question answering, books etc. In difference to textual instructions, these corpora contains relatively rich texts with complex sentence structure. This sometimes results in the inability of the parser to assign the correct POS-tags to the textual instructions, resulting in turn in the inability of the system to correctly identify actions in the texts.

Proceedings of Recent Advances in Natural Language Processing, pages 831–838,
Varna, Bulgaria, Sep 4–6 2017.
https://doi.org/10.26615/978-954-452-049-6_106

To address this problem, we propose a simple rule-based model that attempts to correct the incorrectly assigned POS-tags. We argue that it is not necessary to collect corpora with textual instructions and use them to train the parser, when our simple rules suffice to significantly improve the action detection.

2 The Challenge in Existing Parsers

Textual instructions are usually in imperative form and have short sentence structure. One would expect that this will improve the performance of the parser when assigning POS-tags and dependencies. This is, however, often not the case, as such sentences start with a verb instead of a noun and the subject is almost always missing. It seems that state of the art parsers are trained on corpora that expect the sentence to start with a subject or an object. This results in the parser (usually) assigning a noun tag to the verb and a verb tag to the noun describing the object in the sentence. This in turn causes two problems in systems that attempt to generate automatic human behaviour models from textual instructions: 1. the system is unable to identify all actions mentioned in the text; 2. the system is unable to identify all objects to which the actions are applied.

Here we define *action* to be a verb in its infinite or present form. *Object*, on the other hand, is either the accusative (direct) object of an action or a noun connected to the verb with a preposition. The latter provides spacial, locational or directional information about the action being executed. It is usually correctly identified from the parser, while the direct object is sometimes falsely identified as verb, especially when the sentence actually starts with a verb and the verb is incorrectly identified as a noun.

For example, let's take the sentence "Water the plants once a day." Table 1 gives an example of the way different parsers tagged this sentence. We compare the TurboParser[1] (TP) developed by the Carnegie Mellon University, the Google parser[2] (GP) which is part of the Google cloud natural language API, and the Stanford parser[3] (SP) developed by the Stanford University. It can be seen that the TurboParser and the Stanford parser annotated the verb as a noun, while the Google parser managed to recognise it as correct.

	water	the	plants	once	a	day
TurboParser	NNP	DT	NNS	RB	DT	NN
Google parser	VB	DT	NNS	RB	DT	NN
Stanford parser	NNP	DT	NNS	RB	DT	NN

Table 1: Comparison between the results from different parsers. VB stands for verb in base form, DT for determiner, NN for noun singular, NNP for proper noun, NNS for noun plural, RB for adverb.

Another example is the sentence "Clean the drawer with warm water." (see Table 2). In this case the Stanford parser was unable to recognise "clean" as verb while the other two parsers were able to correctly assign the tag.

	clean	the	drawer	with	warm	water
TurboParser	VB	DT	NN	IN	JJ	NN
Google parser	VB	DT	NN	IN	JJ	NN
Stanford parser	NNP	DT	NN	IN	JJ	NN

Table 2: Comparison between the results from different parsers. VB stands for verb in base form, DT for determiner, NN for noun singular, IN for preposition, JJ for adjective, NNP for proper noun.

And yet another example is "Season to taste with salt and set aside." (see Table 3). Here, the Google parser and the TurboParser were unable to recognise "season" as verb. Interestingly enough, the TurboParser annotated "set" as verb in past tense which will also exclude it as an action.

	season	to	taste	with	salt	and	set	aside
TP	NN	TO	VB	IN	NN	CC	VBD	RB
GP	NN	TO	VB	IN	NN	CC	VB	RB
SP	VB	TO	VB	IN	NN	CC	VB	RB

Table 3: Comparison between the results from different parsers. VB stands for verb in base form, NN for noun singular, IN for preposition, TO for to, CC for conjunction, VBD for verb in past tense, RB for adverb.

The above examples stand to show that at some point each of the parsers incorrectly assigns verb tags to sentences in textual instructions. One option to solve this problem would be to collect a corpus with textual instructions, annotate it and use it to adjust the model of the parser. This is however a time-consuming and error-prone process as it require an expert to annotate all the collected data. In what follows, we argue that it suffices to use simple rules to post-process the POS-tagged text, which already significantly improves

[1] http://demo.ark.cs.cmu.edu/parse

[2] https://cloud.google.com/natural-language/

[3] http://nlp.stanford.edu:8080/parser/index.jsp

the ability of a system to identify actions in textual instructions. We concentrate on the Stanford parser as it is one of the most popular parsers for english language. The rules can, however, be applied to any parser as all parsers seem to produce the same type of errors.

3 A Simple Model for Improving the Stanford Parser

To address the above problem we propose three rules, which can be used in order to (attempt to) correct the assigned by the parser tags.

The first rule is the "no verb rule". It states that if there are no verbs in the sentence (according to the POS-tags assigned by the parser), then we replace the tag of the first word in the sentence with a verb tag (see Rule 1).

Rule 1 ***(No verb rule)*** *Given a sentence S such that $S := (W, T)$ where $W := (w_1, \ldots, w_n)$ is the set of words in S and $T := (t_1, \ldots, t_n)$ the corresponding POS-tags with n being the order of appearance of the word in S. The "no verb rule" states that if $\not\exists\, t \in T$ with $t :=$ verb, then we replace $t_1 :=$ verb.*

Here the meaning of "verb" includes base form verbs, singular verb in present both third person and non-third person, and plural verb in present tense. We use this rule because it is unlikely that there will be no verb in a sentence. In that case we assume that the parser incorrectly annotated the verb as something else and we make a guess that the verb should be the first word in the sentence. This addresses the problem we observed in Table 1 where the Stanford parser and the TurboParser did not annotate any of the words as a verb.

The second rule is the "past tense rule" and it states that if the first word in a sentence is a past tense verb, it should be replaced with a present tense verb (see Rule 2).

Rule 2 ***(Past tense rule)*** *Given a sentence S such that $S := (W, T)$ where $W := (w_1, \ldots, w_n)$ is the set of words in S and $T := (t_1, \ldots, t_n)$ the corresponding POS-tags with n being the order of appearance of the word in S. If $t_1 := VBD$, where VBD denotes past tense verbs, then replace $t_1 := VBD$ with $t_1 :=$ verb.*

We introduce this rule because sometimes the parser assigns the wrong tense to the verb (especially with irregular verbs) and it could lead to the system being unable to find an action (i.e. present tense verb) in the sentence. This case illustrates the error the TurboParser did in Table 3 with the verb "set". Here "set" is not at the beginning of the sentence but some works propose separating sentences not only on full stop but also on conjunctions as such words usually indicate two instructions combined in one sentence.

The third rule is the "noun rule" and it states that if the first word in the sentence is a noun, then most probably something is wrong. In case the second word is a verb, then the tags are exchanged, so that the first word is a verb and the second is a noun. In case the second word is not a verb, then only the tag of the first word is changed to verb (see Rule 3).

Rule 3 ***(Noun rule)*** *Given a sentence S such that $S := (W, T)$ where $W := (w_1, \ldots, w_n)$ is the set of words in S and $T := (t_1, \ldots, t_n)$ the corresponding POS-tags with n being the order of appearance of the word in S. If $t_1 :=$ noun and $t_2 :=$ verb, then assign $t_1 :=$ verb and $t_2 :=$ noun. If $t_1 :=$ noun and $t_2 := \neg$verb, assign $t_1 :=$ verb.*

This rule is used in case the parser has mistakenly annotated the verb as a noun and then predicted that the next word should be a noun. The second case of this rule is the more common one and the problem is also observed in Table 3 where the Google parser annotated the verb at the beginning of the sentence as a noun but where Rule 1 will not apply as there is another verb in the sentence.

4 Evaluation

In the previous section we presented the three simple rules for improving the performance of the parser when annotating textual instructions. In what follows we evaluate the approach by applying it on different kinds of textual instructions such as recipes, manuals, how to articles etc.

4.1 Experimental Setup

To evaluate the model, we collected 61 textual instructions consisting of recipes, manuals, and how to instructions for everyday activities and physical exercises. Table 4 shows the type of instructions in the corpus and the number of instructions from a given type.

The mean number of sentences in the corpus was 18 sentences with a maximum of 111 sentences and a minimum of 3 sentences. The mean number of words per sentence was 10 words with a maximum of 21 words and a minimum of 3 words.

type of instructions	# of instructions
experiments descriptions	4
manuals	15
recipes	10
exercises	10
how-to instructions	22

Table 4: Types of instructions in the corpus.

The mean number of actions per sentence was 1.5 actions.

Table 5 shows an example of a textual instruction used for the evaluation.

```
Take a reasonably large plant pot.
Take a bag of soil.
Open the bag of soil.
Pour some soil into the plant pot.
Take the packet of seeds.
Open the packet of seeds.
Take one of your seeds from the packet
and bury it into the soil, but make sure
it's not too deep.
Repeat this with a couple of other seeds,
leaving some distance between the seeds.
Take your plant pot to somewhere that it
will get enough sunlight, like a window
or a green house.
Pour a bit of water as described on
the packet of seeds.
```

Table 5: Excerpt from instruction describing how to plant a seed, taken from http://www.wikihow.com.

The actions in the corpus were then annotated manually by a human expert. This annotation was used as a ground truth. POS-tags were then assigned to the corpus using the Stanford NLP parser[4]. The parsed text was then post-processed with the three rules we proposed. The rules were implemented in Haskell. To evaluate the ability of the Stanford parser and the rules to correctly annotate actions in textual instructions, we implemented a program in Haskell, which extracts verbs in infinitive form and present tense in a sentence. We used the program to extract the actions in the corpus processed by the Stanford parser and once again after our post-processing step.

To evaluate whether the rules improve the performance of the Stanford parser we calculated how accurate the parser can identify actions, as well as the accuracy after applying the rules. We also calculated the precision and the false discovery rate.

[4] http://nlp.stanford.edu:8080/parser/index.jsp

The precision gives us the fraction of correctly discovered actions from all discovered actions. The false discovery rate gives us the fraction of incorrectly discovered actions from all discovered actions.

Finally, we used a paired t-test to calculate whether the difference in performance when using the rules is significant compared to only using the Stanford parser.

To evaluate whether the rules also affect non-instructional texts, we collected the abstracts from 10 scientific papers from the field of artificial intelligence. The mean length of senses in the corpus was 7.7 with a maximum of 14 and minimum of 3 senstences. The mean number of words in the corpus was 25.5 with a maximum of 36 and minimum of 18 words.

Like in the corpus with instructions, we let the Stanford parser annotate the corpus and then repeated the procedure with our rules. We then compared the POS-tags to see whether the rules changed the tags in the corpus.

4.2 Results

The performance when using only the Stanford parser and when using the rules can be seen in Figure 1. It can be seen that applying the rules

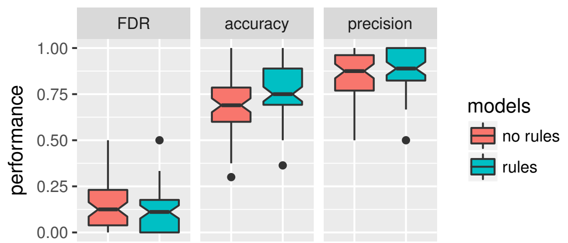

Figure 1: Accuracy, precision, and false discovery rate (FDR) when using the rules (green) and when using only the Stanford parser (red).

increased the accuracy of the recognised actions in the instructions (0.753) compared to using only the Stanford parser (0.687). Furthermore, the notches in the boxplots show the confidence interval around the median. It can be seen that the notches for the accuracy plots do not overlap. This indicates strong evidence that the medians of the results with and without rules differ. Similar trend is seen in the precision (mean precision of 0.864 when using the rules and 0.852 without the rules) and the false discovery rate, where we observe reduced FDR when using the rules. Although in this case the difference in the mean precision and FDR is not large, in Figure 1 it can be seen that the variance when using the rules is much smaller. This indicates that the rules improve the discovery of actions for instructions, in which the Stanford parser performs poorly, but they do not improve the performance for instructions where the parser already performs adequately.

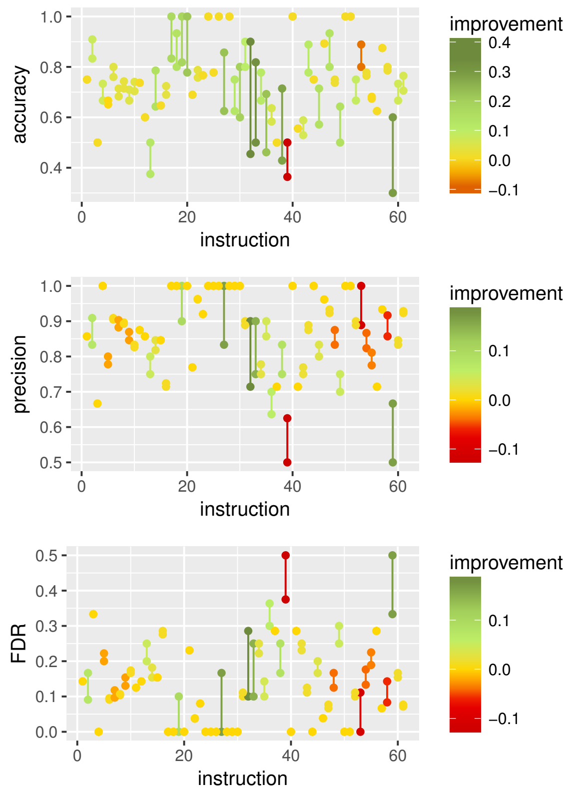

Figure 2: Change in performance when applying the rules. The darker green, the more improvement in the performance when using the rules. Yellow indicates no or little improvement, while red indicates negative improvement.

This is also observed in Figure 2. There the horizontal axis shows the different instructions (from 1 to 61), while the vertical axis shows the accuracy. Furthermore, the vertical lines show the improvement in the performance after applying the rules. Yellow dots indicate no or little improvement, while red indicates negative improvement. Then the darker the green colour, the more improvement is observed when using the rules. It can be seen that there are only two red lines in the accuracy plot, which indicates that in general the rules did not reduce the performance of the Stanford parser. Regarding the precision and FDR, there are 6 instructions with negative improvement but only two of them have less than 5% (the same two instructions as in the accuracy plot). There are some instructions in which no improvement was observed. Here, it has to be pointed out that about half of the instructions, which did not show improvement, already had an accuracy of 100%. The rest showed improvement of up to 40% in the accuracy, 20% in precision and FDR when using the rules. In other words, generally the rules bring positive effect to the recognition of actions in the textual instructions.

Furthermore, we used a paired t-test to see if the results are significantly different when using the rules. The results indicated that the accuracy is significantly different when using the rules (with p-value of 2.09×10^{-6}). This means that the rules provide significant improvement in the performance of the parser when POS-tagging the verbs describing actions in textual instructions.

Finally, applying the rules to the second dataset, which contains abstracts from scientific papers, showed no change in the accuracy of the Stanford parser. In other words, the rules did not influence the ability of the Stanford parser to annotate longer and more complex texts. This stands to show that the rules will probably have no influence on other types of texts that do not follow the structure typical for textual instructions.

5 Discussion

The results from the experiment we conducted showed that it is possible to improve the performance of the Stanford parser when POS-tagging verbs in textual instructions without the need of collecting training data and training the parser's model.

It has to be mentioned, however, that the rules are useful only for situations in which the texts contain short sentences usually in imperative form. It is obvious that the rules will not work for narrative texts with complex sentence structure as such sentences usually do not begin with a verb.

In the same time, the preliminary results on longer texts showed that the rules do not have any influence on non-instructional texts. It is possible that there are situations, in which the rules will influence complex texts, but we do not believe there will be a significant decrease in the performance of the POS-tagger.

We also observed that the rules (usually) do not improve the POS-tagging to a degree where all actions in a text are discovered, which indicates that there are also other factors influencing the performance of the POS-tagger. A more thorough analysis of these factors could provide better understanding of the problem and potentially a basis for more powerful rules. Regardless, the results showed that the three simple rules we proposed significantly improve the recognition of actions in textual instructions. In the same time they save us the time and effort needed to collect and annotate data for training the model of the Stanford parser.

To some, this problem occurring in a relatively small subtype of texts could be insignificant. We, however, believe this is a problem that should be addressed as there is an increasing number of works that rely on natural language instructions to generate models of human behaviour (Yordanova, 2017b; Preum et al., 2017; Lindsay et al., 2017). These models are then used for various applications such as planning, robot manipulation, behaviour recognition of people with impairments. The latter application domain is especially sensitive as the inability to identify possible actions could result in the system being unable to detect potentially dangerous and health threatening behaviour.

In that sense, the improvement of state of the art parsers would also improve the understanding of human behaviour described in natural language.

6 Related Work

To our knowledge, there are no works addressing the specific problem of annotating textual instructions. There are, however, different works that address improving the POS-tagging for various other application domains. For example, (Owoputi et al., 2013) propose using unsupervised clustering methods to improve the POS-tagging accuracy in online conversational texts like Twitter texts. They were able to improve the accuracy from 90% to 93%. In this case however, they still need large corpus for clustering. Furthermore, conversational texts have different structure from textual instructions and the results shown in the paper suggest that state of the art parsers are already able to accurately annotate the texts.

(Ferraro et al., 2013) propose adapting the POS parsers with domain specific rules in order to improve the parser performance. They, however, concentrate on medical narratives which contain a specific structure and terminology relevant only for the medical domain. Similarly, works from (Dickinson, 2006; Kubler et al., 2010) propose adjusting the domain knowledge and POS-tag set for a specific domain instead the annotation algorithm itself.

(Manning, 2011), one of the researchers behind the Stanford parser, also argues that there is little improvement in the parser one could obtain from better machine learning or better features for a classifier. He suggests that the largest opportunity for improvement is improving the descriptive linguistics of the corpus. He points out that many errors arise from the lack of the meaning of the word in the training corpus or simply by errors in the ground truth. He suggests analysing the problems and using automated rules to correct the tags. In that sense, our approach follows this idea and defines rules for the specific domain of textual instructions.

7 Conclusion and Future Work

In this work we presented a simple model consisting of three rules, that attempts to correct the incorrectly annotated verbs or nouns in textual instructions. We showed that this simple approach is already able to improve the discovery of actions in the text. We argued that this was achieved with only three simple rules and significantly reduced the time and effort associated with providing a training corpus for a specific domain.

Although the approach is not always able to provide 100% discovery of the actions in the instructions, the empirical results showed that it does not reduce the quality of the original annotation. In other words, we could only benefit from it. This in turn improves the ability of behaviour understanding systems relying on natural textual descriptions to generate more accurate models of human behaviour. This can potentially contribute to the better understanding of the general problem of structured behaviour representation and the learning of complex behaviour models.

In this work we tested the rules only on the Stanford parser. It is possible that they will have different effect when using other parsers. We do not believe this to be the case, as our observations with other parsers showed that they make similar errors. Nevertheless, in the future it is worth conducting a systematic evaluation of the effect of the rules on different POS-taggers. We also plan to collect a larger dataset with textual instructions as well as random texts in order to better evaluate the effect of the rules on the POS-tags.

In the future, we also intend to conduct a more thorough analysis of the types of errors associated with instructions in natural language. We intend to extend the model to include rules addressing all parts of speech and not only the actions (verbs respectively). In this context, we intend to add rules for correcting the dependencies between words in sentences as they are important for identifying semantic relations when generating human behaviour models (Yordanova, 2017a).

We also plan to investigate the mechanisms for annotating pro-drop languages and controlled languages and potentially adapt them for our problem domain.

We hope that improving the performance of the POS-tagger will ultimately considerably improve the performance of systems that generate models of human behaviour from textual instructions.

Acknowledgments

This work is funded by the German Research Foundation (DFG) within the context of the project TextToHBM, grant number YO 226/1-1.

References

S. R. K. Branavan, Luke S. Zettlemoyer, and Regina Barzilay. 2010. Reading between the lines: Learning to map high-level instructions to commands. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '10, pages 1268–1277. http://dl.acm.org/citation.cfm?id=1858681.1858810.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, pages 740–750. http://www.aclweb.org/anthology/D14-1082.

Markus Dickinson. 2006. An investigation into improving part-of-speech tagging. In *Proceedings of the Third Midwest Computational Linguistics Colloquium (MCLC-06)*. Urbana-Champaign, IL.

Jeffrey P Ferraro, Hal Daume III, Scott L DuVall, Wendy W Chapman, Henk Harkema, and Peter J Haug. 2013. Improving performance of natural language processing part-of-speech tagging on clinical narratives through domain adaptation. *Journal of the American Medical Informatics Association* 20(5):931–939. https://doi.org/10.1136/amiajnl-2012-001453.

Laura M. Hiatt, Anthony M. Harrison, and J. Gregory Trafton. 2011. Accommodating human variability in human-robot teams through theory of mind. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*. AAAI Press, Barcelona, Spain, IJCAI'11, pages 2066–2071. https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-345.

Frank Krüger, Kristina Yordanova, Veit Köppen, and Thomas Kirste. 2012. Towards tool support for computational causal behavior models for activity recognition. In *Proceedings of the 1st Workshop: "Situation-Aware Assistant Systems Engineering: Requirements, Methods, and Challenges" (SeASE 2012) held at Informatik 2012*. Braunschweig, Germany, pages 561–572.

Sandra Kubler, Matthias Scheutz, Eric Baucom, and Ross Israel. 2010. Adding context information to part of speech tagging for dialogues. In *Proceedings of the 2010 International Workshop on Treebanks and Linguistic Theories*. Tartu, Estonia.

A. (Alan) Lindsay, J. (Jonathon) Read, J. F. (Joo) Ferreira, T. (Thomas) Hayton, J. (Julie) Porteous, and P. J. (Peter) Gregory. 2017. Framer: Planning models from natural language action descriptions. In *Proceedings of the 27th International Conference on Automated Planning and Scheduling (ICAPS)*. Pittsburg, USA. http://hdl.handle.net/10149/620942.

Christopher D. Manning. 2011. Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? In *Proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text Processing - Volume Part I*. Springer-Verlag, Berlin, Heidelberg, CICLing'11, pages 171–189. http://dl.acm.org/citation.cfm?id=1964799.1964816.

André F. T. Martins, Noah A. Smith, Eric P. Xing, Pedro M. Q. Aguiar, and Mário A. T. Figueiredo. 2010. Turbo parsers: Dependency parsing by approximate variational inference. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Stroudsburg, PA, USA, EMNLP '10, pages 34–44. http://dl.acm.org/citation.cfm?id=1870658.1870662.

Tuan A Nguyen, Subbarao Kambhampati, and Minh Do. 2013. Synthesizing robust plans under incomplete domain models. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, Curran Associates, Inc., pages 2472–2480. http://papers.nips.cc/paper/5120-synthesizing-robust-plans-under-incomplete-domain-models.pdf.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A

Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Atlanta, GA.

S. M. Preum, A. S. Mondol, Meiyi Ma, Hongning Wang, and J. A. Stankovic. 2017. Preclude: Conflict detection in textual health advice. In *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. pages 286–296. https://doi.org/10.1109/PERCOM.2017.7917875.

Miquel Ramirez and Hector Geffner. 2011. Goal recognition over pomdps: Inferring the intention of a pomdp agent. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*. AAAI Press, Barcelona, Spain, volume 3 of *IJCAI'11*, pages 2009–2014. https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-335.

Kristina Yordanova. 2016. From textual instructions to sensor-based recognition of user behaviour. In *Companion Publication of the 21st International Conference on Intelligent User Interfaces*. ACM, New York, NY, USA, IUI '16 Companion, pages 67–73. https://doi.org/10.1145/2876456.2879488.

Kristina Yordanova. 2017a. Automatic generation of situation models for plan recognition problems. In *Proceedings of Recent Advances in Natural Language Processing*. Varna, Bulgaria. Accepted.

Kristina Yordanova. 2017b. TextToHBM: A generalised approach to learning models of human behaviour for activity recognition from textual instructions. In *Proceedings of the AAAI Workshop on Plan, Activity and Intent Recognition (PAIR)*. AAAI, San Francosco, USA, pages 891–898. https://www.aaai.org/ocs/index.php/WS/AAAIW17/paper/view/15110.

Kristina Yordanova and Thomas Kirste. 2015. A process for systematic development of symbolic models for activity recognition. *ACM Transactions on Interactive Intelligent Systems* 5(4):20:1–20:35. https://doi.org/10.1145/2806893.

Kristina Yordanova and Thomas Kirste. 2016. Learning models of human behaviour from textual instructions. In *Proceedings of the 8th International Conference on Agents and Artificial Intelligence (ICAART 2016)*. Rome, Italy, pages 415–422. https://doi.org/10.5220/0005755604150422.

Ziqi Zhang, Philip Webster, Victoria Uren, Andrea Varga, and Fabio Ciravegna. 2012. Automatically extracting procedural knowledge from instructional texts using natural language processing. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*. European Language Resources Association (ELRA), Istanbul, Turkey, pages 520–527. ACL Anthology Identifier: L12-1094. http://www.lrec-conf.org/proceedings/lrec2012/pdf/244_Paper.pdf.