Robust processing of noisy web-collected data

Jelke Bloem
University of Groningen
j.bloem.3@student.rug.nl

Michaela Regneri
Saarland University
regneri@coli.uni-saarland.de

Stefan Thater
Saarland University
stth@coli.uni-saarland.de

Abstract

Crowdsourcing has become an important means for collecting linguistic data. However, the output of web-based experiments is often challenging in terms of spelling, grammar and out-of-dictionary words, and is therefore hard to process with standard NLP tools. Instead of the common practice of discarding data outliers that seem unsuitable for further processing, we introduce an approach that tunes NLP tools such that they can reliably clean and process noisy data collected for a narrow but unknown domain. We demonstrate this by modifying a spell-checker and building a coreference resolution tool to process data for paraphrasing and script learning, and we reach state-of-the-art performance where the original state-of-the-art tools fail.

1 Introduction

Web experiments in general and crowdsourcing in particular have become increasingly popular as sources for linguistic data and their annotations. For annotation purposes, crowdsourcing seems to deliver high-quality results (Snow et al., 2008) even for complex tasks like resolving anaphora (Chamberlain et al., 2009) or semantic relations (von Ahn et al., 2006). Collecting textual raw data via crowdsourcing seems very appealing at first sight, because it gives access to large amounts of commonsense knowledge which can usually not be extracted from text (Regneri et al., 2010; Singh et al., 2002).

However, processing the output of crowdsourcing or any web-based experiment (like corpora with emails, blog posts or Twitter messages) poses major challenges for natural language processing (NLP) tasks that take such data as their input: The data usually contains a lot of spelling and grammar mistakes and many idiosyncratic words. Most domain-independent standard tools such as part of speech taggers, parsers or coreference resolution systems are not robust enough to handle such unpredictable idiosyncrasies. It is possible to use manual annotation instead of NLP tools to process web-collected datasets (Vertanen and Kristensson, 2011; Regneri et al., 2010), however, this does not scale well to larger experiments and is thus not always an option.

Many web-based approaches rely on collecting multiple instances for a narrow task or domain, and thus validate any text by comparing it to other examples from the same experiment and filtering them out if they seem too idiosyncratic (Law and von Ahn, 2009). We consider this approach too restrictive, because it may discard instances that are actually good but simply rare, or discard too much data when many instances are noisy. We instead make use of similar instances in crowdsourced data at a later stage in the processing pipeline: We show that we can enhance the output of standard NLP systems rather than immediately discarding the original texts. Instead of restricting the input set to a much smaller, more homogeneous resource that is easier to process, we want to keep as much variety as possible in the data, by putting more knowledge into the actual systems. Such data parallelism can also be found in paraphrasing data or comparable corpora, but we are not aware of tool adaptation in this area.

We show how to implement that paradigm by example of two NLP tools, namely spell checking and coreference resolution. This paper is structured as follows: We first introduce a highly parallel dataset of web-collected event sequence descriptions for several common-sense tasks in the

kitchen domain (Sec. 2). The main focus of the remainder is the preprocessing of the data, using tools modified as described above, such that it can be put into a previously introduced paraphrasing pipeline. We explain and evaluate our modification of an open-source spell-checker (Sec. 3), and we demonstrate that a coreference resolution technique that solely relies on the parallel dataset can outperform a state-of-the-art system or a system trained on a general corpus (Sec. 4).

2 A highly parallel data set

Scripts (Schank and Abelson, 1977) describe a certain scenario (e.g. "eating in a restaurant") with temporally ordered events (*the patron enters restaurant, he takes a seat, he reads the menu...*) and participants (*patron, waiter, food, menu,...*). Written event sequences for a scenario have been collected by Regneri et al. (2010). A similar collection is used by Rohrbach et al. (2012), with basic kitchen tasks from *Jamie Olivers Home Cooking Skills*[1]. They collect natural language sequences using Mechanical Turk. For each scenario, subjects were asked to provide tutorial-like sequential instructions for executing the respective kitchen task. Overall, the corpus contains data for 52 cooking tasks, with 30-50 sequences per task. Fig. 1 shows three examples for the scenario *preparing garlic*.

Making Use of Redundancy

We mainly use the kitchen data set collected by Rohrbach et al. (2012). The data was gathered for script mining, which involves finding "event paraphrases" within the text, e.g. determine that *chop them finely* and *chip the garlic up until its small* denote the same event. In order to match event paraphrases with many variants in wording, spelling correction (for all kinds of processing) and pronoun resolution (to compare the phrase adjuncts) are essential (e.g. to match *chop them* with the misspelled phrase *chip the garlic up*).

Regneri et al. (2011) have already shown how to re-train a parser to gain robustness on the bullet-point style sentences by modifying big training corpora so as to replicate the nonstandard syntax. We want to show how the smaller data set itself can serve to modify or filter standard applications in order to gain robustness and process the noisy data reliably. Given the example in Fig. 1, it is clear that big parts of the content word inventory (*cloves*, *skin*, *garlic*) are shared between the three sequences. Under this prerequisite, we propose two different application modifications for preprocessing:

1. **Spelling Correction:** We use the data to amend the lexicon of a standard spell-checker. The kitchen data set contains many domain specific words, e.g. *microplane grater*, which the standard spell-checker corrects to *micro plane grater*. We also use the vocabulary of the input texts to rank correction proposals of actual misspellings.

2. **Coreference Resolution:** Coreference resolution is a hard unsolved problem, even for standard text. Our fragmentary discourses that consist of incomplete sentences were not processable by any state-of-the-art tool: On top of the problematic syntax, the systems fail because the most salient referent is sometimes not even present in the text itself. E.g. under the headline *Preparing garlic*, one can give a complete instruction without using the word *garlic* by referring to it with pronouns only. Given the very restricted set of possible noun antecedents in our data, selectional preferences can be used to fully resolve most pronominal references.

3 Spelling correction task

Orthographically correct text is much more reliable as input for standard applications or dictionaries, thus we need to spell-check the noisy web texts. We use the widely used spelling correction system GNU Aspell[2]. Aspell's corrections are purely based on the edit- and phonological distance between suggestions and the misspelled token, and many kitchen-domain specific words are not in Aspell's dictionary. This leads to implausible choices that could easily be avoided by considering the domain context — due to the parallel nature of our data set, correct words tend to occur in more than one sequence. The first objective is to gain more precision for the *identification*

[1] http://www.jamieshomecookingskills.com/

[2] http://www.aspell.net/

1.	first strip of the papery skin of the bulb	1.	peal the skin of a clove of garlic	1.	remove the papery skin of the garlic clove
2.	ease out as many intact cloves as possible	2.	cut the ends of of the clove	2.	mash the glove into pulp
3.	chop them finely if you want a stronger taste	3.	chip the garlic up until its small	3.	this can be done using a ziploc bag and a meat tenderizer
4.	chope them coarsely if you want a weaker taste	4.	use in your favorite dish	4.	use the garlic pulp in all sorts of meals
5.	crushed garlic is the strongest taste				

Figure 1: Three example sequences for the scenario *preparing garlic*

of misspellings. We indirectly expand Aspell's dictionary: If a word occurs in at least n other sequences in the same spelling, the system accepts it as correct. (In our experiment, $n = 3$ turned out to be a reliable value.) As a second enhancement, we improve *correction* by guiding the selection of corrected words. We verify their plausibility by taking the first suggested word that occurs in at least one other sequence. This excludes out-of-domain words that are accidentally too similar to the misspelled word compared to the correct candidate. Similar work on spelling correction has been done by Schierle et al. (2008), who trained their system on a corpus from the same domain as their source text to gain context. We take this idea a step further by training on the noisy (but redundant) dataset itself, which makes definition and expansion of the domain superfluous.

Evaluation

We compare our system's performance to a baseline of standard Aspell, always picking the first correction suggestion. Since error detection is handled by standard Aspell in both cases, we have not evaluated recall performance. Previous work with manual spelling correction showed low recall (Regneri et al., 2010), so Aspell-based checking is unlikely to perform worse. Fig. 3 shows a comparison of both methods using various measures of precision, based on manual judgement of the corrections made by the checkers. Since Aspell's spelling error detection is not perfect, some of the detected errors were not actually errors, as shown in the manually counted "False Positives" column. For this reason, we also included "true precision", which is calculated only over actual spelling errors. Another measure relevant for the script learning task is "semantic precision", a more loosely defined precision measure in which any correction that results in any inflection of the desired lemma is considered correct, ignoring grammaticality. The numbers show that we gain 15% overall precision, and even 22% by considering genuine misspellings only. If we relax the measure and take every correct (and thus processable) lemma form as correct, we gain an overall precision of 18%.

4 Pronoun resolution task

The pronoun resolution task involves finding pronouns, looking for candidate antecedents that occur before the pronoun in the text, and then selecting the best one (Mitkov, 1999). While our dataset adds complications for standard pronoun resolution systems, the domain simplifies the task as well. Due to the bullet point writing style, first person and second person pronouns always refer to the subject performing a task. This leaves only third person pronouns, of which possessives are uninteresting for the script learning task — with just one person active in the dataset, there should be no ambiguity attributable to personal possessive pronouns (Regneri et al., 2011). There is also a relatively restricted space of possible antecedents due to the short texts.

Our system uses a POS-tagged and dependency-parsed version of the dataset, created using the Stanford parser (De Marneffe et al., 2006). The last few noun phrases before the pronoun within the same sequence are considered candidate antecedents, as well as the main theme of the descriptive scenario title. It then uses a model of selectional preference (Clark and Weir, 2001) to choose the most likely antecedent, based on verb associations found in the rest of the data. For example, in *Grab a knife and slice it*, the fact that *it* is the object of *slice* can be used to estimate that *knife* is an unlikely antecedent, given that such a verb-object combination does not occur elsewhere in the data.

Our approach only includes the noisy dataset in its selectional preference model, rather than some external corpus. For the phrase *chop it*, the statist-

Method	Precision	False Positives	True Precision	Sem. Precision	Corrections
Aspell	0.43	0.28	0.57	0.58	162
Enhanced Aspell	**0.58**	0.29	**0.79**	**0.76**	150

Figure 2: Evaluation of the baseline and improved spell-checker on 10 scenarios (25.000 words).

ical association strength of candidate antecedents with the head *chop* is checked, as computed on our dataset. The candidate that is most strongly associated with *chop* is then proposed as the preferred antecedent. Association values are computed with the marginal odds ratio measure on a dictionary of all head-subject and head-object relations occurring in our data. While this model is computationally much more simple, it takes advantage of our parallel data.

Evaluation

We compare the models to two baselines: One is a state-of-the-art general, openly available pronoun resolver based on Expectation Maximalization (Charniak and Elsner, 2009). This system relies on grammatical features to model pronouns. Therefore it fails to model the pronouns in our parsed noisy data in many cases (recall 0.262). For the other simpler models, it is trivial to achieve full recall on 3rd person pronouns in our parsed sequences, so we won't further evaluate recall. We instead evaluate the models in terms of correctness compared to human annotation. As a second baseline, we use a state-of-the-art vector space model (Thater et al., 2011) to compute selectional preferences instead of tuning the preferences on our dataset. For each word or phrase, this model computes a vector representing its meaning, based on a large general corpus. For each candidate antecedent of phrases containing a pronoun like *chop it*, we compute the model's vector for the original phrase in which the pronoun is instantiated with the antecedent (*chop garlic*, *chop fingers*, ...). The candidate vector that is closest to the vector of the original verb, computed as the scalar product, is taken as the preferred antecedent.

The results show that our approach outperforms both baselines, despite its simpler heuristics. Compared to the vector space model approach we observe a 16% correctness gain, with a much larger gain over the EM system due to its recall issue.

Model	Correct
Vector space model	0.544
EM	0.175
Odds ratio	**0.631**

Figure 3: Evaluation of different selectional preference models for pronoun resolution, on two scenarios (103 pronouns total).

5 Conclusion and Future work

We have demonstrated a new approach to processing the generally noisy output of crowdsourcing experiments in a scalable way. By tuning NLP tools on the dataset they will process, we have shown performance gains on two NLP tasks - spell-checking and coreference resolution. Despite using relatively simple methods and heuristics compared to the state of the art, the parallel nature of our dataset allowed us to achieve state-of-the-art performance. Our approach to processing preserves more crowdsourced information than the common practise of discarding outlier data. Results could be improved further by refining the methods, e.g. weighting candidate antecedents by their distance to the pronoun in the pronoun resolution tool, or improving detection of common spelling mistakes by varying how easily unknown words are accepted based on edit distance. In future research, applying this approach to other datasets and other processing tasks would be interesting, as well as extending the process to languages that lack state-of-the-art NLP tools.

Acknowledgements

We thank Manfred Pinkal, Gosse Bouma and the anonymous reviewers for their helpful comments on the paper.

References

Jon Chamberlain, Massimo Poesio, and Udo Kruschwitz. 2009. A demonstration of human computation using the phrase detectives annotation game. In *KDD Workshop on Human Computation*.

E. Charniak and M. Elsner. 2009. EM works for pronoun anaphora resolution. In *Proceedings of EACL*, pages 148–156. Association for Computational Linguistics.

S. Clark and D. Weir. 2001. Class-based probability estimation using a semantic hierarchy. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, pages 1–8. Association for Computational Linguistics.

M.C. De Marneffe, B. MacCartney, and C.D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.

Edith Law and Luis von Ahn. 2009. Input-agreement: a new mechanism for collecting data using human computation games. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 1197–1206, New York, NY, USA. ACM.

R. Mitkov. 1999. Anaphora resolution: The state of the art. *Unpublished Manuscript*.

M. Regneri, A. Koller, and M. Pinkal. 2010. Learning script knowledge with web experiments. In *Proceedings of ACL-10*.

Michaela Regneri, Alexander Koller, Josef Ruppenhofer, and Manfred Pinkal. 2011. Learning script participants from unlabeled data. In *Proceedings of RANLP 2011*, Hissar, Bulgaria.

Marcus Rohrbach, Michaela Regneri, Micha Andriluka, Sikandar Amin, Manfred Pinkal, and Bernt Schiele. 2012. Script data for attribute-based recognition of composite activities. In *Computer Vision - ECCV 2012 : 12th European Conference on Computer Vision*, volume 2012 of *Lecture Notes in Computer Science*, Firenze, Italy, October. Springer, Springer.

Roger C. Schank and Robert P. Abelson. 1977. *Scripts, Plans, Goals and Understanding*. Lawrence Erlbaum, Hillsdale, NJ.

M. Schierle, S. Schulz, and M. Ackermann. 2008. From spelling correction to text cleaning–using context information. *Data Analysis, Machine Learning and Applications*, pages 397–404.

Push Singh, Thomas Lin, Erik T. Mueller, Grace Lim, Travell Perkins, and Wan L. Zhu. 2002. Open mind common sense: Knowledge acquisition from the general public. In *On the Move to Meaningful Internet Systems - DOA, CoopIS and ODBASE 2002*, London, UK. Springer-Verlag.

R. Snow, B. O'Connor, D. Jurafsky, and A.Y. Ng. 2008. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of EMNLP*, pages 254–263. Association for Computational Linguistics.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2011. Word meaning in context: A simple and effective vector model. In *Proc. of IJCNLP 2011*.

K. Vertanen and P.O. Kristensson. 2011. The imagination of crowds: conversational aac language modeling using crowdsourcing and large data sources. In *Proceedings of EMNLP*.

Luis von Ahn, Mihir Kedia, and Manuel Blum. 2006. Verbosity: a game for collecting common-sense facts. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, New York, NY, USA. ACM.