A Taxonomy for In-depth Evaluation of Normalization for User Generated Content

Rob van der Goot, Rik van Noord, Gertjan van Noord
University of Groningen
Broerstraat 5, Groningen,
r.m.van.der.goot@rug.nl, r.i.k.van.noord@rug.nl, g.j.m.van.noord@rug.nl

Abstract

In this work we present a taxonomy of error categories for lexical normalization, which is the task of translating user generated content to canonical language. We annotate a recent normalization dataset to test the practical use of the taxonomy and read a near-perfect agreement. This annotated dataset is then used to evaluate how an existing normalization model performs on the different categories of the taxonomy. The results of this evaluation reveal that some of the problematic categories only include minor transformations, whereas most regular transformations are solved quite well.

Keywords: normalization, user generated content, social media, error taxonomy

1. Introduction

For other natural language processing tasks, such as grammatical error correction and machine translation, there already exist detailed error taxonomies, which help in getting insights in the strengths and weaknesses of systems. For normalization, such an evaluation does not exist yet. Reynaert (2008) proposed an evaluation framework which evaluates the different sub-tasks in more detail; enabling the evaluation of error detection, candidate generation and candidate ranking. In the more recent shared task on lexical normalization hosted at the WNUT workshop (Baldwin et al., 2015b), the outputs of systems were evaluated on precision, recall and F1 score. Orthogonal to these approaches, we propose a more in-depth evaluation of normalization, focusing on categories of different normalization replacements.

Existing error taxonomies are unfortunately not suitable for the task of normalization, since the categories are substantially different. For machine translation, taxonomies as the Multidimensional Quality Metrics (Mariana, 2014) are proposed, which contains 3 main categories: accuracy, verity and fluency. The last category would be the most relevant for normalization, but the normalization task compromises a different variety of errors and anomalies. In grammatical error correction, often a more detailed taxonomy for errors is used; the default benchmark has 28 categories (Ng et al., 2014). However, many of the errors in this taxonomy are not annotated in the normalization benchmarks and many normalization replacements are not included in this taxonomy.

Different benchmarks for normalization specify the task slightly different; one striking example is the inclusion of the expansions of phrasal abbreviations (e.g. 'lol'↦'laughing out loud'). However, this might not be the desired output, since one might argue that this expansion does not represent the intended meaning. This reveals another potential use for a taxonomy of normalization actions: it enables us to filter the categories before training, and thus learn a model which only handles the desired categories.

2. Normalization

There is ample of previous work on normalization, but there is no consensus about the scope of the normalization task. Some existing corpora with normalization annotation for English are shown in Table 1. The corpora by Baldwin and Li (2015) and Kaljahi et al. (2015) are also annotated with error categories. However, the guidelines for the annotation of these corpora are substantially different compared to the other, more commonly used, corpora. The taxonomy proposed by Baldwin and Li (2015) has a very high percentage of normalizations since it allows deletion and insertion of tokens as well as the correction of capitalization. In contrast, The Foreebank (Kaljahi et al., 2015) has a very low percentage of normalized words. This is due to its more canonical forum domain; it is mostly focused on grammatical error correction.

In the rest of this section, we will discuss two examples from these datasets to give a clearer idea of the task:

(1) most social pple r troublesome
most social people are troublesome

Example 1 shows an example tweet from the LexNorm2015 (Baldwin et al., 2015b) corpus and it's annotated normalization. This example includes two subsequent words which are shortened by omitting vowels.

(2) i aint messin with no1s wifey yo lol
i ain't messing with no one's wifey you laughing out loud

Example 2 includes two examples of 1-n normalization replacements; the first replacement is not only a split, '1s' is also expanded to 'one's' in the annotation. The second 1-n replacement is the expansion of the phrasal abbreviation of 'lol'. The word 'wifey' is kept unchanged, this reflects the conservativeness which is encouraged in the annotation guidelines.

Source	Name	Punct.	Caps.	1-n n-1	Words	%normalized
Yang and Eisenstein (2013)	LexNorm1.2	-	-	-	10,576	11.5
Li and Liu (2014)		-	+-	-	40.560	10.5
Baldwin et al. (2015b)	LexNorm2015	-	-	+	73,806	9.1
Baldwin and Li (2015)		+	+	+	11,890	28.7
Kaljahi et al. (2015)	Foreebank	+	+	+	15,595	3.3

Table 1: Properties of different annotation guidelines. Punct. and Caps.: if respectively punctuation and capitalization is corrected. +- means capitalization is simply copied from the original utterance. The '1-n n-1' column indicates if normalization corrections beyond the word level are allowed.

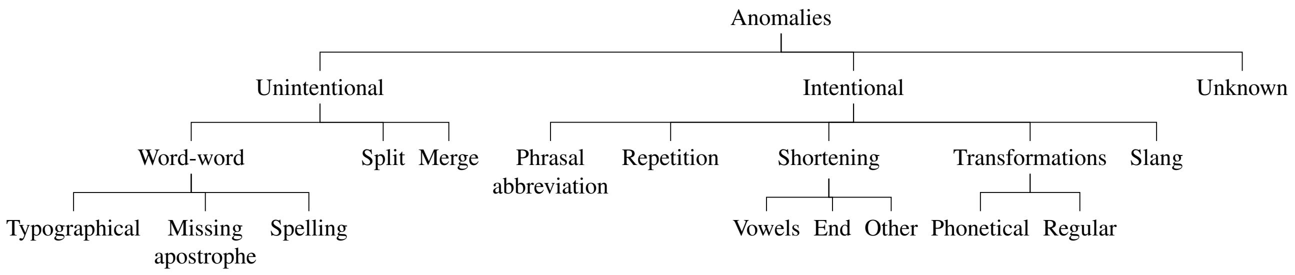

Figure 1: Our proposed taxonomy of anomalies in user generated text.

3. Proposed Taxonomy

Our proposed taxonomy is loosely based on the categories used by the Foreebank (Kaljahi et al., 2015) and Baldwin and Li (2015). Furthermore we took categories from the annotation guidelines of LexNorm2015 (Baldwin et al., 2015a), since they include which kind of anomalies should be annotated. The categories of our taxonomy are a combination of the previously used categories, but are empirically refined during the early stages of annotation. We make a main distinction between intentional and unintentional anomalies since they have a different origin; meaning they might require different handling in NLP systems. Note that we do not include word reordering and capitalization, since these phenomena are not annotated in our dataset. Our taxonomy is shown in Figure 1; accompanying examples can be found in Table 2. We will now describe each final category in more detail.

1. Typographical error This includes small errors, which are a result of mistyping keys on keyboards. In case of doubt with another category, we put words with a character edit distance of one in this category (e.g. bidge$\mapsto$bridge, feela$\mapsto$feels).

2. Missing apostrophe In social media text, the apostrophe is often skipped. Even though this category is relatively trivial to solve, it might have large effects in a pipeline approach, since it can resolve tokenization issues.

3. Spelling error This category includes all cases in which a word is unintentionally used in the wrong form/context, including spelling and grammatical errors. We also include mismatches between American and British English here. When in doubt with the first category, annotators should answer the following question: if the sender were to send the message again, would he/she make the same mistake?

4. Split When a word is split into multiple words. There is one case in our corpus where this happens intentionally ('l o v e'$\mapsto$love), this is still annotated in this category.

5. Merge There is no space between two different words, this is a special case of a typograpical error.

6. Phrasal abbreviation In some datasets, phrasal abbreviations, such as 'lol', 'idk' and 'brb' are expanded to respectively 'laughing out loud', 'I don't know' and 'be right back'. These abbreviations consist of all first characters of the words they represent.

7. Repetition In social media, extra focus is put on words by character repetition. Repetition can also occur on a character N-gram level, e.g. 'hahahahahaha'. Even when adding only one extra character, we categorize the replacement here. If this category collides with a phrasal abbreviation (e.g. lolololol), we choose to categorize it as phrasal abbreviation, since this is more defining for the intended meaning.

8. Shortening vowels A simple way to shorten words is to leave out vowels. In this category we also place words in which most of the vowels are removed, e.g. pple$\mapsto$people.

9. Shortening end Another way to shorten words is too leave out the last syllable(s) or character(s). Based on context, it is often trivial for humans to understand which ending is intended. If the anomaly includes a suffix to indicate plurality, we still classify it in this category (e.g. favs$\mapsto$favorites).

10. Shortening other There are other variations to shorten words. For example, using only the first letter of each part of a compound, skipping another syllable then the last or using standard abbreviations (pdx$\mapsto$portland). This category also contains combinations of the previous two categories (talkn$\mapsto$talking, smth$\mapsto$something).

Category	Examples
1. Typographical error	spirite↦spirit, complaing↦complaining, throwg↦throw
2. Missing apostrophe	im↦i'm, yall↦y'all, microsofts↦microsoft's
3. Spelling error	favourite↦favorite, dieing↦dying, theirselves↦themselves
4. Split	pre order↦preorder, screen shot↦screenshot
5. Merge	alot↦a lot, nomore↦no more, appstore↦app store
6. Phrasal abbreviation	lol↦laughing out loud, pmsl↦pissing myself laughing
7. Repetition	soooo↦so, weiiiiird↦weird
8. Shortening vowels	pls↦please, wrked↦worked, rmx↦remix
9. Shortening end	gon↦gonna, congrats↦congratulations, g↦girl
10. Shortening other	cause↦because, smth↦something, tl↦timeline,
11. Phonetic transformation	hackd↦hacked, gentille↦gentle, rizky↦risky
12. Regular transformation	foolin↦fooling, wateva↦whatever, droppin↦dropping
13. Slang	cuz↦because, fina↦going to, plz↦please
14. Unknown	skepta↦sunglasses, putos↦photos

Table 2: Examples of normalization pairs for each category.

11. Phonetic transformation In an effort to shorten texts phonetic transformations are often used. In this case one or multiple characters are converted to their literal pronunciation to form a word. This can be done with numbers as well as letters. Other cases of phonological transformation occur when people replace characters with similar sounding characters, e.g. s↦z, c↦k d↦t. Note that transformations of word endings like -er↦-a (e.g. brotha↦ brother) fit in the next category, as the normalization is pronounced differently compared to the original word.

12. Regular transformation For this category, we consider common transformations of endings of words. On Twitter it is common to end participles and gerunds with 'in' instead of 'ing'. Another common transformation is to replace the last syllable with 'a'. Note that transformations like cuz↦because does not fit in this category, because this transformation is not transferable to other words.

13. Slang This category includes all other transformations as well as novel words specific to this domain.

14. Unk Annotator is not sure in which category a word belongs. This can be because the annotator does not agree with the normalization annotation, or because the Tweet is not understandable for the annotator.

4. Annotation

To test our proposed taxonomy, we annotated the LexNorm2015 dataset (Baldwin et al., 2015b) with an extra layer, which indicates for each normalization replacement to which category in our taxonomy it belongs. We choose to annotate this dataset because it is publicly available, the most recent, the largest and annotation is verified by shared task participants. It should be noted however, that as long as alignment is available, the taxonomy can easily be adopted for other corpora.

To ease the annotation effort, we annotate unique normalization replacement pairs. Since most ambiguity problems should already be solved by the normalization layer, this is a safe generalization. However, annotators still have access to the contexts, in case of doubt. Sometimes a replacement

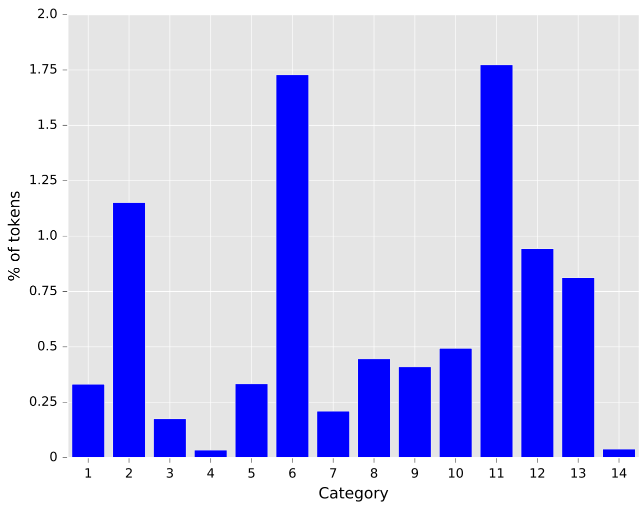

Figure 2: The distribution of the different categories. The percentages of tokens occuring in the categories with respect to all words in the corpus.

fits in two categories, e.g. difffffff↦different (fits in category 7 and 9). In these cases it is up to the annotator to decide which category defines the replacement most. We only annotate the training part of the dataset, to keep the test data strictly for the final testing of a tuned model (note that there is no default development split). The train data can be used in a k-fold cross validation experiment to inspect the strengths and weaknesses of a model.

One annotator annotated all the 1,204 replacement pairs present in the training part of the Lexnorm2015 dataset. Additionally, a second annotator annotated a random shuffle of 150 replacements to test the inter-annotator agreement. All annotators are guided by the descriptions in Section 3. The annotators reached a Cohen's Kappa (Cohen, 1960) of 0.807, which indicates a near perfect agreement. There was no clear trend in the disagreements; the most common disagreement was between category 13 and 8, but this only occured three times. After annotation both annotators discussed and resolved the differently annotated pairs

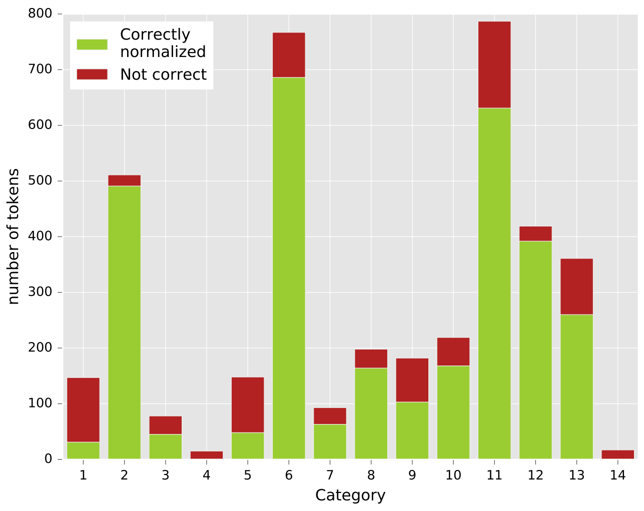

Figure 3: Performance of MoNoise on the different categories in a 10-fold experiment on the training part of LexNorm2015.

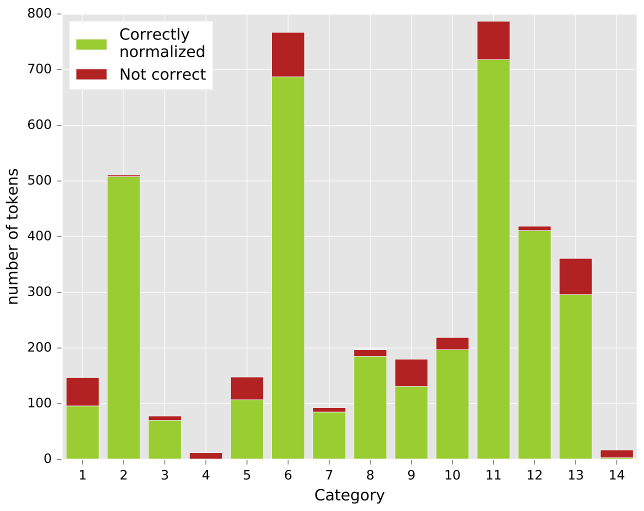

Figure 4: Performance of MoNoise on the different categories when using gold error detection.

and refined the description of the categories.
Figure 2 shows the distribution of the different categories. Most of the replacements are intentional word-word replacements (7-12); about half of these are phonetic transformations. Other popular categories are phrasal abbreviations and missing apostrophe.

5. Evaluation of Normalization Model

In this section we evaluate the normalization model MoNoise (van der Goot and van Noord, 2017) in more detail using our proposed taxonomy. This system is based on the idea that the normalization task consists of different normalization replacement actions. MoNoise generates normalization candidates for all words, and includes the original word as a candidate, so that it can decide whether or not to normalize when the candidates are ranked. It generates candidates using the out-of-the-box spelling correction system Aspell[1], combined with a word embeddings module and a lookup list generated from the training data. The candidates are then ranked in a random forest classifier using features from the generation modules combined with N-gram probabilities from a canonical dataset as well as a Twitter dataset. For more details we refer to the original paper (van der Goot and van Noord, 2017).
For maximum performance we train MoNoise using the Aspell bad-spellers mode. We run MoNoise in a 10-fold cross validation setup to get predictions for the whole training set. The absolute number of correctly normalized as well as the missed transformations are plotted in Figure 3. Note that besides these errors, 198 canonical words are wrongly normalized, so this still accounts for a large part of the errors. Two categories are not handled at all by MoNoise, namely split (4) and unknown (14). Besides these, the most difficult categories are typographical erros (1), phonetic transformations (11) and merge (5). Surprisingly, two of these categories consists of a lot of minor transformations (1, 11). Replacements beyond the word level are clearly difficult; a closer look at the typographical errors revealed that in these cases the original token is often ranked first by the normalization model. The categories that are almost completely resolved are missing apostrophe (2) and regular transformations (12). For missing apostrophe this is due to the relative restricted number of different replacements, whereas for regular transformations this is due to the fact that they are clearly in need of normalization, and their correct counterparts are quite similar (often in$\mapsto$ing).
Figure 4 shows the performance per category when using gold error detection. The performance is substantially higher on almost all categories, confirming that error detection is still a relevant problem. Unsurprisingly, the categories split (4) and unknown (14) do not improve much, since the correct candidates are not generated by MoNoise, also the phrasal abbreviation category does not improve. This is because these are only solved by using the lookup list; this module does not improve by using gold error detection. Categories which previously performed well (2, 12) are now almost completely solved.

6. Conclusion

We proposed a taxonomy for normalization replacements, which can be used for detailed evaluation of normalization systems as well as the filtering of training data. While some categories can potentially overlap, the inter-annotator agreement indicates near perfect agreement. We tested a state-of-art normalization model to see which categories are still most problematic: typographical errors, phonetical errors and merging. On most of the other categories the system performs quite well. When using gold error detection, the performance goes up for almost all categories, indicating that this is still a far from solved problem. Interesting future work would be an extrinsic evaluation: how important are the different normalization categories for specific downstream tasks.
The annotation as well as the used scripts are available at: `https://bitbucket.org/robvanderg/normtax`

[1] `http://www.aspell.net`

Acknowledgements

We would like to thank Barbara Plank and Antonio Toral for discussions about the taxonomy. This work is part of the Parsing Algorithms for Uncertain Input project, funded by the Nuance Foundation.

Bibliographical References

Baldwin, T. and Li, Y. (2015). An in-depth analysis of the effect of text normalization in social media. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 420–429, Denver, Colorado, May–June. Association for Computational Linguistics.

Baldwin, T., de Marneffe, M.-C., Han, B., Kim, Y.-B., Ritter, A., and Xu, W. (2015a). Guidelines for English lexical normalisation. Technical report, Workshop on Noisy User-generated Text.

Baldwin, T., de Marneffe, M.-C., Han, B., Kim, Y.-B., Ritter, A., and Xu, W. (2015b). Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 126–135, Beijing, China, July. Association for Computational Linguistics.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Kaljahi, R., Foster, J., Roturier, J., Ribeyre, C., Lynn, T., and Le Roux, J. (2015). Foreebank: Syntactic analysis of customer support forums. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1341–1347, Lisbon, Portugal, September. Association for Computational Linguistics.

Li, C. and Liu, Y. (2014). Improving text normalization via unsupervised model and discriminative reranking. In *Proceedings of the ACL 2014 Student Research Workshop*, pages 86–93, Baltimore, Maryland, USA, June. Association for Computational Linguistics.

Mariana, V. R. (2014). The Multidimensional Quality Metric (MQM) framework: A new framework for translation quality assessment. *The Journal of Specialised Translation*.

Ng, H. T., Wu, S. M., Briscoe, T., Hadiwinoto, C., Susanto, R. H., and Bryant, C. (2014). The CoNLL-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14, Baltimore, Maryland, June. Association for Computational Linguistics.

Reynaert, M. (2008). All, and only, the errors: more complete and consistent spelling and ocr-error correction evaluation. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may. European Language Resources Association (ELRA).

van der Goot, R. and van Noord, G. (2017). MoNoise: Modeling noise using a modular normalization system. *Computational Linguistics in the Netherlands Journal*, 7.

Yang, Y. and Eisenstein, J. (2013). A log-linear model for unsupervised text normalization. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 61–72, Seattle, Washington, USA, October. Association for Computational Linguistics.