Abbreviation Explorer - an interactive system for pre-evaluation of Unsupervised Abbreviation Disambiguation

Manuel R. Ciosici
UNSILO A/S and
Aarhus University
Aarhus, Denmark
manuel@cs.au.dk

Ira Assent
Department of Computer Science
Aarhus University
Aarhus, Denmark
ira@cs.au.dk

Abstract

We present *Abbreviation Explorer*, a system that supports interactive exploration of abbreviations that are challenging for Unsupervised Abbreviation Disambiguation (UAD). *Abbreviation Explorer* helps to identify long-forms that are easily confused, and to pinpoint likely causes such as limitations of normalization, language switching, or inconsistent typing. It can also support determining which long-forms would benefit from additional input text for unsupervised abbreviation disambiguation. The system provides options for creating corrective rules that merge redundant long-forms with identical meaning. The identified rules can be easily applied to the already existing vector spaces used by *UAD* to improve disambiguation performance, while also avoiding the cost of retraining.

1 Introduction

Abbreviations are short forms of concepts that authors employ to avoid repeated typing of long text sequences (e.g. *National Aeronautics and Space Administration* is abbreviated to *NASA*). Because long-forms are represented by shorter sequences of characters (often just two or three), even in the same domain, multiple concepts may map to the same short-form. Such ambiguous abbreviations, i.e., short-forms with several potential meanings, are quite common. A study by Liu et al. (2001) showed that 23.1% of abbreviations in the Unified Medical Language System (UMLS) ontology, a popular resource in the field of medicine, are ambiguous, meaning that they map to more than one long-form.

Abbreviation disambiguation is the task of identifying the intended long-form for an ambiguous short-form, given its use in a sentence. Expansion and disambiguation of abbreviations can make technical text easier to read (Ciosici and Assent, 2018). The task of disambiguating abbreviations in context has been included in both the 2013 and 2014 ShAReCLEF eHealth Challenge (Mowery et al., 2016).

Unsupervised Abbreviation Disambiguation (UAD) (Ciosici et al., 2019) is a recent unsupervised approach that identifies ambiguous abbreviations, and makes use of word embeddings to identify the intended long-forms given contextual uses of ambiguous abbreviations. While successfully disambiguating the vast majority of the ambiguous abbreviations, some difficult cases remain. Following an idea of pre-evaluation in suggested by Ciosici et al. (2019), we present a system called *Abbreviation Explorer* that supports investigation and correction of word embedding spaces learned by *UAD*. Highlighting difficult cases, it allows inspection of likely causes and potential resolution through rewrite rules. The rules can either be used to retrain *UAD*, or directly applied to its vector space, thus eliminating the need to retrain the model. *Abbreviation Explorer* does not target simple variations in long-forms such as plurals, hyphenation, or minor text variations as identified by Zhou et al. (2006); Moon et al. (2015). Such noise is eliminated by *UAD*'s pre-processing pipeline. Rather, it focuses on semantically challenging abbreviations, or those where the unstructured text input corpus provided to *UAD* does not provide sufficient context for successful disambiguation. *Abbreviation Explorer* is a general tool for long-form normalization that does not require expert knowledge or domain-specific, human-curated knowledge bases that approaches like Melton et al. (2010) rely on.

Abbreviation Explorer thus supports the user in understanding which abbreviations are difficult to disambiguate, why *UAD* might not have been able to learn how they differ, in issuing corrections that immediately improve the disambiguation perfor-

Proceedings of NAACL-HLT 2019: Demonstrations, pages 1–5
Minneapolis, Minnesota, June 2 - June 7, 2019.

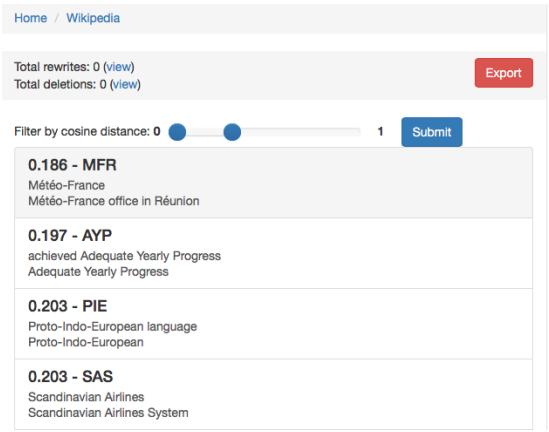

Figure 1: The main screen of *Abbreviation Explorer* listing long-forms that are challenging for disambiguation.

mance, or in determining where more input text would allow learning of better representations and result in improved disambiguation. All this, without having to conduct expensive, large-scale evaluation on abbreviation disambiguation tasks that require manually labeled data. In fact, *Abbreviation Explorer* takes as input only the vector model trained by *UAD* and the training data that was used to generate the vector model. [1]

2 Pre-evaluation analysis

Long-forms that map into the same short-form and are close to each other in the *UAD* vector space are correlated with low disambiguation performance (Ciosici et al., 2019). Long-forms end up close to each other in the vector space due to incorrect long-form normalization in *UAD*, inconsistent reference by humans (*annual average daily traffic* vs. *average annual daily traffic*), language switching (*Centre National de la Recherche Scientifique* vs. *Centre for Scientific Research*), multi-layer abbreviations (*Voice over IP* vs. *Voice over Internet Protocol*), organization name changes (*Organisation of Islamic Cooperation* vs. *Organisation of the Islamic Conference*), or lack of sufficient examples in the input text for a proper representation.

Abbreviation Explorer identifies these cases using the cosine distance between pairs of long-forms belonging to the same short-form, and presents them to the user for interactive exploration. Figure 1 shows the main screen of *Abbreviation Explorer* displaying a list of long-forms and

[1] A presentation video of the system is available at https://youtu.be/XsBb7QMkDdg

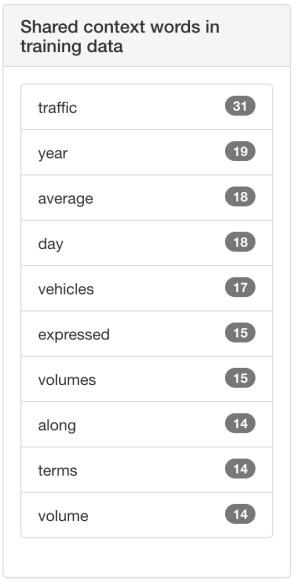

Figure 2: Detail screen with subset of words that appear often in the context of the long-forms *average annual daily traffic* and *annual average daily traffic*. The large frequency of contextual terms indicates that the two long-forms likely denote the same concept.

Figure 3: Detail screen with subset of words close to the two challenging long-forms. Red rows indicate words that are close to both long-forms. The large overlap indicates that the two long-forms likely denote the same concept.

the cosine distance between them. In some cases, it is obvious even without domain knowledge that the pairs are lexical variations that should be collapsed into a single long-form.

For less obvious cases, *Abbreviation Explorer* provides the user with a page containing information that supports investigation. It provides two indicators for contextual analysis: a list of most common words observed in training examples for both long-forms, and a list of the top 30 words in the word embedding space that are close to each long-form. The list of common words observed in training data for both long-forms helps identify cases where the two long-forms denote the same concept, or where the training data is inconclusive for effective disambiguation. The list of top words in the vector space can be used to pinpoint long-forms that denote the same concept even when the training data does not contain significant word overlap. Figures 2 and 3 show the two views for the abbreviation *AADT* which is expressed in the data as *annual average daily traffic* and *average annual daily traffic*. The view helps users conclude that the two long-forms are term variations of the same concept as they are used in similar contexts. On the other hand, long-forms that denote separate concepts often have little, if any, overlap between the sets of close words. In Figure 4, we can see that for the abbreviation *ABC*, its two correctly identified long-forms *American Broadcasting Company* and *Australian Broadcasting Corporation* have no shared context as they denote separate concepts.

Abbreviation Explorer supports two kinds of actions for pairs of close long-forms that should be corrected. The user can either choose to issue a manual rewrite rule, forcing the system to collapse one long-form into the other, or ask for a long-form to be deleted from the system. The latter action is useful for addressing issues in the input text processing that are not captured in *UAD*'s pre-processing. For example, in the case of the short-form FN, *Front* is picked up as a long-form instead of *Front Nationale* (which is the correct long-form) due to the greedy nature of *UAD*'s normalization which identifies Front as a long-form because it contains both letters *F* and *N*.

Rewrite and deletion rules from *Abbreviation Explorer* can be exported as a *JSON* document which can then be used in two ways. It can either be applied to the training data employed by *UAD*,

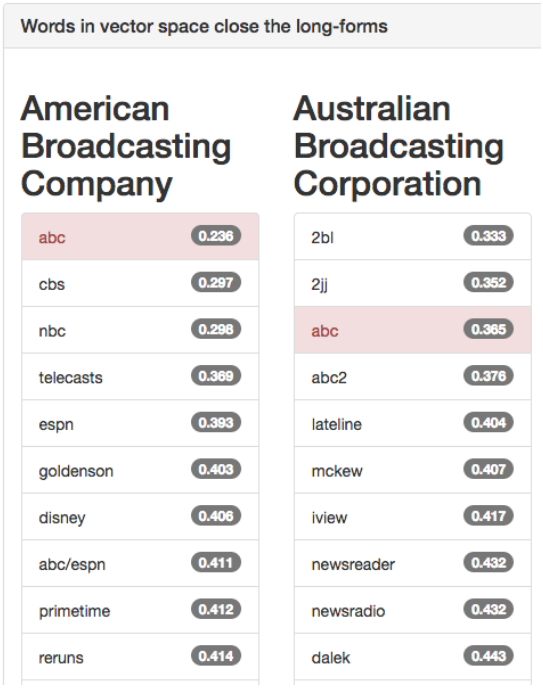

Figure 4: Detail screen with subset of words in the vector space close to the two challenging long-forms. Red rows indicate words that are close to both long-forms. The lack of vocabulary overlap indicates that the two long-forms denote separate concepts.

and then be used to train another word vector space, or they can be applied directly to the word embedding space to adjust vectors of long-forms. Using the correction rules directly on *UAD*'s vector space avoids the effort required to re-derive word vector spaces. In the next section we discuss the performance benefits of the various ways of applying corrections to *UAD*.

3 Evaluation of Abbreviation Explorer

In order to study the effect of manual corrections created using *Abbreviation Explorer*, we the same Wikipedia data set that was used in the evaluation of *UAD* (Ciosici et al., 2019). For all experiments, we employed $10 - fold$ cross-validation, with the same folds used in the *UAD* evaluation. In experiments requiring *UAD* training, we used the same *word2vec* hyper-parameters as in *UAD* evaluation.

With *Abbreviation Explorer*, we identified 40 pairs of long-forms that denote the same concept. As mentioned earlier, correction rules can either be applied to the data followed by a retraining of *UAD*, or they can be applied directly to already trained *UAD* models. We study two ways to correct preexisting vector spaces: (1) for every pair of long-forms denoting the same concept remove

			Weighted			Macro		
	Disambiguator	Acc.	Prec.	Rec.	F1	Prec.	Rec.	F1
1	UAD with TXT	94.28	96.17	94.28	94.76	90.84	93.29	90.98
2	Removed long-forms	**96.38**	**97.34**	**96.38**	**96.61**	**92.72**	**95.15**	**93.15**
3	Averaged long-form vectors	96.37	97.32	96.37	96.60	92.64	95.12	93.11
4	Retrained UAD with TXT	96.28	97.33	96.28	96.54	92.64	95.13	93.07

Table 1: Comparisons with UAD on data set Wikipedia (no-stopwords).

one long-form from the vector space and (2) for every pair of long-forms denoting the same concept, we replace the two long-form vectors with their average. In Table 1, we provide a comparison of disambiguation performance following the three corrective methods. The first row contains disambiguation performance of *UAD* without any corrections; rows 2 and 3 contain the two vector adjustment methods applied to the vector space from the first row using corrections identified with *Abbreviation Explorer*; and the last row contains the performance of *UAD* retrained on data where we applied corrections identified with *Abbreviation Explorer*.

All three corrective methods outperform *UAD* on the original data, using no corrections, by up to 1.85 weighted F1 points. Both precision and recall are improved after applying the corrective rules. This illustrates the usefulness of *Abbreviation Explorer* in supporting users to issue manual corrections for difficult long-forms. Another important result in Table 1, is that the three correction methods studied result in performances within 0.07 weighted F1 points of each other. Removing a long-form from the vector space, or averaging two long-forms is considerably less costly than a complete re-derivation of the word embedding space utilized by *UAD*. This leads us to conclude that adjusting *UAD*'s vector spaces is preferred as it results in similar performance to retraining *UAD*, but does not incur the time cost of re-deriving word vector spaces.

4 Conclusion

We present *Abbreviation Explorer*, a system that supports interactive exploration of abbreviations that are easily confused by *Unsupervised Abbreviation Disambiguation (UAD)*. *Abbreviation Explorer* works by identifying and understanding long-forms whose learned word embedding representations are close to each other and providing contextual information to help users understand which long-forms denote the same concept. *Abbreviation Explorer* can assist users who are not domain-experts in generating corrective rules that address abbreviation disambiguation difficulties due to incorrect long-form normalization, inconsistent typing, language switching, multi-layer abbreviations, organization name changes, and more. Corrective rules generated using *Abbreviation Explorer* can be used to improve the disambiguation performance of *UAD*, even without performing expensive full-scale evaluation using labeled data. The corrective rules can be applied to the already existing vector spaces used by *UAD* to improve disambiguation performance, while, at the same time, avoiding the cost of retraining *UAD*.

References

Ciosici, Manuel, Sommer, Tobias, Assent, and Ira. 2019. Unsupervised Abbreviation Disambiguation Contextual disambiguation using word embeddings. *arXiv e-prints*, page arXiv:1904.00929.

Manuel Ciosici and Ira Assent. 2018. Abbreviation expander-a web-based system for easy reading of technical documents. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 1–4.

Hongfang Liu, Yves A Lussier, and Carol Friedman. 2001. A study of abbreviations in the UMLS. In *Proc. AMIA Symp.*, page 393.

Genevieve B Melton, SungRim Moon, Bridget McInnes, and Serguei Pakhomov. 2010. Automated identification of synonyms in biomedical acronym sense inventories. In *Proc. Louhi Workshop on Text and Data Mining of Health Documents*, pages 46–52.

Sungrim Moon, Bridget McInnes, and Genevieve B Melton. 2015. Challenges and Practical Approaches with Word Sense Disambiguation of Acronyms and Abbreviations in the Clinical Domain. *Healthc Inform Res*, 21(1):35–42.

Danielle L Mowery, Brett R South, Lee Christensen, Jianwei Leng, Laura-Maria Peltonen, Sanna

Salanterä, Hanna Suominen, David Martinez, Sumithra Velupillai, Noémie Elhadad, Guergana Savova, Sameer Pradhan, and Wendy W Chapman. 2016. Normalizing acronyms and abbreviations to aid patient understanding of clinical texts: ShARe/CLEF eHealth Challenge 2013, Task 2. *Journal of Biomedical Semantics*, 7(1):43.

Wei Zhou, Vetle I. Torvik, and Neil R. Smalheiser. 2006. Adam: another database of abbreviations in medline. *Bioinformatics*, 22(22):2813–2818.