Cleaning Up After a Party: Post-processing Thesaurus Crowdsourced Data

Oksana Antropova[1(✉)], Elena Arslanova[1], Maxim Shaposhnikov[1], Pavel Braslavski[1,2,3], and Mikhail Mukhin[1]

[1] Ural Federal University, Yekaterinburg, Russia
choksy@mail.ru, contilen@gmail.com, chelseamax@yandex.ru, {pavel.braslavsky,mikhail.mukhin}@urfu.ru
[2] JetBrains Research, Saint Petersburg, Russia
[3] National Research University Higher School of Economics, Saint Petersburg, Russia

Abstract. The study deals with post-processing of a noisy collection of synsets created using crowdsourcing. First, we cluster long synsets in three different ways. Second, we apply four cluster cleaning techniques based either on word popularity or word embeddings. Evaluation shows that the method based on word embeddings and existing dictionary definitions delivers best results.

Keywords: Crowdsourcing · Thesaurus · Semantic resources

1 Introduction

Thesauri and wordnets are widely used in various natural language processing tasks and applications. There are several approaches to creating wordnets: manual building by professional lexicographers; automatic construction from text corpora and semi-structured sources such as Wiktionary; as well as approaches based on crowdsourcing. In the latter case, non-professional volunteers or paid workers collaboratively construct thesaurus in small steps. Each of the approaches has its pros and cons. For example, crowdsourcing allows quick generation of data at a low cost, but this data is usually noisy and needs post-processing.

In this work, we address the task of analyzing and processing the data generated within the YARN (*Yet Another RussNet*) project [3]. YARN has a web interface[1] allowing virtually everyone to edit existing synsets or create new ones. Manual analysis of these crowdsourced synsets shows that the collection is quite noisy: synsets are duplicated (several non-identical synsets correspond to a concept), may contain irrelevant word entries (synsets often mix synonyms,

O. Antropova, E. Arslanova and M. Shaposhnikov contributed equally to the paper.
[1] https://russianword.net/editor.

D. Ustalov et al. (Eds.): AINL 2018, CCIS 930, pp. 133–138, 2018.
https://doi.org/10.1007/978-3-030-01204-5_13

hypernyms and other semantically related words), or are incomplete. Still the synsets created by volunteers have their advantages – they include vocabulary not presented in traditional dictionaries, such as recent borrowings, multi-word expressions, or vulgar words.

In this study, we experiment with several methods aimed at post-processing and cleaning YARN synset collection. First, we cluster long synsets in three different ways. Second, we apply four cleaning techniques based either on word popularity or word embeddings. Evaluation shows that the methods based on embeddings provide better results and can be applied to YARN data.

2 Related Work

Initially, thesauri have been created manually by professional linguists and lexicographers [5]. Crowdsourcing became a viable option for creation and expansion of linguistic resources since its inception in the mid-2000s [6]. For example, [1] describes an experiment on creating a sense inventory using MTurk platform.

YARN project [3] aims at creating an open thesaurus of the Russian language using crowdsourcing. Its user interface [4] allows creating synsets with some guidance from dictionary data. A rather relaxed user action control leads to quite noisy results. A related study [7] describes deduplication of YARN synsets. The authors concluded that three overlapping word entries is an optimal threshold for merging crowdsourced synsets. Cluster cleaning methods using word embeddings employed in the current study are close to synset induction/sense disambiguation methods, see for example an overview in [10]. The difference of the methods herein is that they use dictionary definitions to disambiguate senses.

3 Data

To date, there are 69,796 synsets and 143,508 entries in YARN [2]. 64.3% of synsets contain one or two words, 33.6% – from three to nine, 2.1% – over nine words. In the current study we worked with 23,408 synsets of length from 3 to 9.

Clustering. Following [7], we clustered the collection of synsets in three ways that we denote *GREEDY*, *TRIPLES*, and *BABENKO*. *GREEDY* is a variant of single linkage clustering – synsets sharing three words are clustered, which results in 16,694 clusters. In case of *TRIPLES* each word triple occurring in the initial synsets defines a cluster. It is a variant of soft clustering: synsets can be assigned to multiple clusters. This process generates 20,966 clusters. The third option makes use of a machine-readable dictionary of Russian synonyms.[3] The dictionary contains 29,194 entries organized into 7,538 synsets. For each YARN synset in the initial collection we searched for the closest dictionary synset in terms of Jaccard coefficient and clustered synsets belonging to the same

[2] https://russianword.net/data.

[3] Babenko, L.G.: The thesaurus dictionary of the Russian language synonyms, 2008.

BABENKO synset together. The rationale behind this methods is to enrich dictionary synsets with crowdsourced multi-word expressions, recent borrowings, etc. *BABENKO* clustering resulted in 9,323 clusters (YARN synsets not linked to *BABENKO* synsets become single-synset clusters).

Collection of Definitions. We also use dictionary definitions to re-organize synset clusters. The majority of definitions employed in the study come from Wiktionary[4]. Missing definitions were collected from dictionaries by Efremova, Ozhegov, Ushakov, and Babenko.[5] In total, the collection comprises 187,003 unique definitions for 132,485 word entries.

Gold Synsets. In order to tune parameters of the methods and evaluate them, we manually created a small collection of 'gold synsets'. We started with 1,140 noisy synsets. First, we manually clustered them in such a way that synsets in a cluster describe the same concept, which resulted in 139 clusters. Second, a gold synset was created for each cluster by removing duplicate and irrelevant words. The gold synsets were randomly split into training set (39) used for parameter tuning and test set (100).

4 Methods

4.1 Words and Synsets Weighing

Two methods exploiting redundancy produced by the crowd were developed for synset cluster cleaning: *WordWeights* and *SynsetRanks*.

WordWeights method is based on a simple idea: the more synsets in the cluster contain the word the more likely it is actually relevant to the concept. Thus, for every word in a cluster, we calculate its weight as a share of initial synsets it occurs in. If the word weight exceeds a threshold, the word is added to the 'pure' synset representing the cluster. The threshold is optimized on the training set.

SynsetRanks aims at estimating the quality of the initial synset as a whole and then compiling a 'clean' synset from good ones. First, synsets in a cluster are ranked based on their weights. Synset weight is calculated as average of its *WordWeights*. Synsets below a threshold are discarded. The remaining synsets are merged incrementally top-down if their similarity exceeds the second threshold. Both parameters of the routine are tuned on the training set.

4.2 Cleaning with Embeddings

The second group of methods is based on word embeddings and collection of dictionary definitions. We employed pre-trained *fasttest* 300-dimensional vectors

[4] https://ru.wiktionary.org/.
[5] An overview of dictionary data available for Russian can be found in [8].

from RusVectōrēs project [9][6]. *Fasttext* vector representation combines word-level and character n-grams embeddings, which is helpful in case of highly inflectional Russian language and partly mitigates out-of-vocabulary problem [2]. Gensim[7] library was used to query the model and calculate cosine word similarities.

Word2vec[8] uses vector representations of words (in case of multiwords a sum of constituents' vectors is used). First, we calculate all pairwise similarities of words in the cluster. Second, we rank words according to the average similarity to all cluster members. Thus, more central words are ranked higher. Then we incrementally build new synsets from the top of the list by adding words if their average similarity to the items already in synset exceeds a threshold.

Def2vec model represents word senses as an average of vectors constituting their definitions. By this approach, each word is represented by a *set* of vectors, each reflecting one of its senses. After initial pre-ordering of the words as in the *Word2vec* model the extension of the classic agglomerative clustering algorithm is performed. The main advantage of the approach is that we account for polysemy; moreover, the newly built synsets are delivered with definitions.

5 Results and Discussion

For every 'golden' cluster we found the closest newly obtained cluster by Jaccard similarity. Then we applied a cleaning method with all the possible parameters to every cluster aligned with the training set and chose the parameters that provided the best results. After that we applied all the methods with the optimized parameters to the test set and evaluated the results, see Table 1.

Table 1. Evaluation results. *J* – Jaccard coefficient, *P* – precision, *R* – recall, *F1* – F1-score.

	GREEDY				TRIPLES				BABENKO			
Method	*J*	*P*	*R*	*F1*	*J*	*P*	*R*	*F1*	*J*	*P*	*R*	*F1*
WordWeights	0.42	0.58	0.63	0.54	0.53	0.62	0.84	0.67	0.48	0.76	0.60	0.60
SynsetRanks	0.40	0.59	0.53	0.51	0.55	0.68	0.80	0.68	0.46	0.70	0.58	0.58
Word2vec	0.51	0.70	0.69	0.65	0.55	0.73	0.74	0.69	0.49	0.68	0.66	0.63
Def2vec	0.45	0.77	0.57	0.60	0.52	0.81	0.63	0.66	0.45	0.68	0.64	0.60

As long as the proposed evaluation method estimates only the clusters aligned with the 'gold' clusters, the results shown in the Table 1 are overestimated to

[6] http://rusvectores.org/ru/models/.
[7] https://radimrehurek.com/gensim/.
[8] We use *word2vec* as a name of a general approach to word embeddings and to contrast it to the latter method that works with definitions.

some extent. Especially strongly it affects the results of *TRIPLES*, because it creates many small clusters (43% of aligned clusters consist of one or two synsets). Simple alignment of completely unprocessed YARN synsets with the 'gold' synsets provides $Jaccard = 0.71$ and $F1 = 0.81$, so it explains why *TRIPLES* delivers the highest values. Nonetheless, we would not recommend using this method, because it is overrated and leaves most of the duplicates unclustered.

GREEDY tends to mix different senses of polysemic words. For example, it unites different senses of the word ***край***: *land, country, territory, side* and *brim*. Success of *WordWeights* and *SynsetRanks* in such cases depends on what concept dominates in YARN (usually this is the most frequent sense). *Word2vec* and *Def2vec* manage well as long as relevant contexts and definitions are found. *BABENKO* clusters are noticeably cleaner and usually contain more synsets, which work best for redundancy-based methods. *Word2vec* and *Def2vec*, in opposite, do not succeed with this clustering, because *BABENKO* clusters usually contain closely related words that have similar definitions and occur in similar contexts. Table 2 illustrates these considerations with the synsets obtained for the concept *state, country*.

Table 2. Results aligned with the 'gold' sysnset {*держава*[1], *государство*[2], *страна*[3]}

Method	GREEDY	BABENKO
Word Weights	*область*[4], *сторона*[5], *территория*[6], *регион*[7], *страна*[3], *государство*[2], *местность*[8], *край*[9]	*держава*[1], *государство*[2], *страна*[3]
Synset Ranks	*область*[4], *местность*[8], *край*[9], *страна*[3], *округа*[10]	*держава*[1], *государство*[2], *страна*[3]
Word2vec	*держава*[1], *государство*[2], *царство*[11], *княжество*[12], *империя*[13], *страна*[3]	*государственная власть*[14], *содружество*[15], *государство*[2], *империя*[13], *страна*[3], *нация*[16], *государственное управление*[17], *держава*[1]
Def2vec	*королевство*[18], *держава*[1]	*государство*[2], *империя*[13], *царство*[11], *нация*[16], *держава*[1], *страна*[3], *королевство*[18], *содружество*[15], *территория*[6]

[1]*(literary)* state, [2]state, [3]country, [4]province, [5]side, [6]territory, [7]region, [8]area, [9]land, [10]neighbourhood, [11]realm, [12]princedom, [13]empire, [14]government, [15]commonwealth, [16]nation, [17]state authorities, [18]kingdom

Despite the fact that redundancy methods demonstrate relatively good results in case of *BABENKO* clustering, in fact they produce only synsets related to the most general concepts (because such words usually dominate), and discard all the data related to more specific concepts. *Word2vec* and *Def2vec* keep

all words from initial YARN synsets intact. *Def2vec* delivers higher precision and additionally provides definitions for newly created synsets. *Word2vec* works better if there are no relevant meanings in the dictionary. For example, it generates a correct synset {***штрих, корректор***} for the concept *correction fluid*, whereas *Def2vec* splits the pair according to their dictionary meanings: *stroke* for ***штрих*** and *corrector* (profession) for ***корректор***.

6 Conclusion

The quality of crowdsoursed YARN synsets varies greatly. Most of them mixes two and more similar concepts and are incomplete at the same time. Nonetheless, as a whole they cover significantly more vocabulary than traditional synonym dictionaries. The proposed methods of post-processing allow to improve average quality of synsets, but they can hardly distinguish synonyms from hyponyms/hypernyms and co-hyponyms. Thus, the study confirms that crowdsourced projects demand well-thought user action control and organization.

Word2vec provides best recall among the examined methods and can be recommended if followed by manual editing by a qualified lexicographer. Otherwise, *Def2vec* delivering highest precision is quite a practical option. We plan to apply *GREEDY* clustering and *Def2vec* cleaning to YARN data.

Acknowledgments. PB was supported by RFH grant #16-04-12019, OA was supported by RFBR according to the research project No. 18-312-00129.

References

1. Biemann, C.: Creating a system for lexical substitutions from scratch using crowdsourcing. Lang. Resour. Eval. **47**(1), 97–122 (2013)
2. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606 (2016)
3. Braslavski, P., Ustalov, D., Mukhin, M., Kiselev, Y.: YARN: spinning-in-progress. In: GWC, pp. 58–65 (2016)
4. Braslavski, P., Ustalov, D., Mukhin, M.: A spinning wheel for YARN: user interface for a crowdsourced thesaurus. In: EACL (demo), pp. 101–104 (2014)
5. Fellbaum, C.: Wordnet: An Electronic Database. MIT Press, Cambridge (1998)
6. Gurevych, I., Kim, J. (eds.): The People's Web Meets NLP. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-35085-6
7. Kiselev, Y., Ustalov, D., Porshnev, S.: Eliminating fuzzy duplicates in crowdsourced lexical resources. In: GWC, pp. 161–167 (2016)
8. Kiselev, Y., et al.: Russian lexicographic landscape: a tale of 12 dictionaries. In: Dialogue, pp. 254–271 (2015)
9. Kutuzov, A., Kuzmenko, E.: Webvectors: a toolkit for building web interfaces for vector semantic models. In: AIST, pp. 155–161 (2017)
10. Ustalov, D., Panchenko, A., Biemann, C.: Watset: automatic induction of synsets from a graph of synonyms. In: ACL, pp. 1579–1590 (2017)