Crowdsourcing Synset Relations with Genus-Species-Match

Dmitry Ustalov*†
*IMM UB RAS, Yekaterinburg, Russia
†Ural Federal University, Yekaterinburg, Russia
dau@imm.uran.ru

***Abstract*—Enabling a domain-specific lexical resource is useful for improving the performance of a natural language processing system. However, such resources may be represented in the form of glossaries—terms provided with their sense definitions. Despite the problem of integrating such domain-specific glossaries into more sophisticated general purpose resources like thesuari being highly topical, it is complicated by ambiguity of the individual terms. This paper presents Genus-Species-Match, a crowdsourcing workflow for matching noisy pairs of synsets representing hyponymic/hypernymic relations. The system demonstrates F1 score of 80% on an experiment conducted on an online labor marketplace using the EMERCOM glossary and the Yet Another RussNet sense inventory.**

I. Introduction

Social media monitoring and opinion mining systems do often use electronic dictionaries and thesauri for improving natural language processing performance by query expansion and entity disambiguation. However, such thesauri often represent common lexis only, which makes it highly topical either to extend them with domain-specific lexicon or even to provide a new resource dedicated to the domain. So far, crowdsourcing seems to be a very promising approach for building various lexical resources including electronic thesauri [1]. Hence, it seems to be reasonable to apply this human-computer technique for transforming such a glossary into an open source thesaurus, which may be used by anyone, who requires processing texts in Russian belonging to the emergency management domain.

For instance, in the emergency management domain it is highly important to classify incoming texts between authorized units, determine the type of emergency, and track the media impact of a particular incident. The EMERCOM of Russia maintains a specialized glossary [2], which contains only terms provided with sense definitions, which are neither grouped into sets of quasi-synonyms (synsets), nor have been provided with hyponymic/hypernymic relations between each other. While it is still possible to align such a glossary to a sense inventory or even a full-fledged thesaurus, the ambiguity problem should be addressed first.

The present work makes the following contributions: (1) it proposes a novel workflow for efficient disambiguation of hyponymic/hypernymic relations between synsets, and (2) presents an approach for transformation of a domain-specific glossary into an electronic thesaurus with synsets and relations between them. In the preliminary experiment, a subset of the EMERCOM glossary has been merged into YARN—an open source sense inventory for Russian made though crowdsourcing [3]. As the result, the proposed system achieved F_1 score of 80%. The deliverables are available under open source licenses making it possible to reproduce the present study.

The rest of this paper is organized as follows. Section II reviews the related work. Section III defines the problem of matching synsets with the corresponding hyponymic/hypernymic relations. Section IV presents an approach for glossary transformation and a crowdsourcing workflow named Genus-Species-Match. Section V describes the experimental setup used for evaluating the workflow. Section VI presents and discusses the obtained results. Section VII concludes with final remarks and directions for the future work.

II. Related Work

The literature review is dedicated to three aspects. Firstly, the automatic thesaurus construction methods will be discussed (see Astrakhantsev & Turdakov [4] for details). Secondly, related crowdsourcing design patterns will be reviewed. Finally, a short summary on popular adaptive crowdsourcing frameworks will be presented.

A. Automatic Thesaurus Construction

Hearst proposed a set of predefined lexico-syntactic patterns for extracting hyponymy relations from unrestricted text corpora [5]. A similar approach has been exploited by Yang & Powers for providing a broader set of semantic relations recognized by a syntactic parser [6]. Giuliano et al. presented an approach for thesaurus construction, which leverages on the Wikipedia structure to extract concepts and terms denoting them using natural language texts and interlinks between them [7]. Hagiwara, Ogawa & Toyama, instead, used the probabilistic latent semantic indexing technique for acquiring synonyms automatically from large corpora [8]. Sarasua, Simperl & Noy created CrowdMap [9], a model for aligning a pair of knowledge resources between each other semi-automatically by verifying algorithmic matching using the tasks posted on an online labor market.

B. Crowdsourcing Workflows

The most recognized crowdsourcing workflow is Find-Fix-Verify developed by Bernstein et al. for Soylent [10], a Microsoft Word plugin that submits human intelligence tasks (HITs) to an online labor marketplace for rephrasing and improving the original text. As its name implies, the workflow includes three stages: (1) on the *Find* stage crowd workers

ISBN 978-952-68397-0-7 (paperback), ISBN 978-952-68397-1-4 (PDF)

should find a text area that can be shortened without changing the meaning, (2) on the *Fix* stage the workers should propose improvements for these text areas, and (3) on the *Verify* stage the workers have to select the *worst* proposed fixes.

Wang et al. presented CrowdER [11], an entity resolution technique that uses hybrid human-machine workflow for duplicate detection and record linkage. CrowdER has two stages: (1) *estimating* likelihood between entity pairs and filtering out those below the specified threshold, and (2) *verifying* the remaining pairs of records.

Biemann proposed Turk Bootstrap Word Sense Inventory (TWSI) [12], a workflow for the creation of a word sense inventory using crowdsourcing. It is based on the lexical substitution phenomenon and is composed of three stages: (1) *finding* substitutions by asking workers to provide synonyms for the highlighted words in sentences, (2) *aligning* senses by assessing the semantic similarity between sentence pairs, and (3) *matching* the meaning by evaluating the derived word senses on usage examples.

Ustalov & Kiselev developed Add-Remove-Confirm [13], a three-stage workflow for improving synset quality: (1) at the *Add* stage, the workers propose the words to be inserted into the synset, (2) at the *Remove* stage, the workers select the words to be removed, and (3) at the final *Confirm* stage, the workers select the best synset variant.

C. Adaptive Crowdsourcing Frameworks

Adaptive crowdsourcing frameworks has gained a significant interest today due to the noisy nature of the crowdsourcing phenomenon. Particularly, in various crowdsourcing applications, the workers' expertise is unknown, which makes it necessary to develop statistical techniques for estimating the workers' reliability and aggregating the submitted answers.

Surprisingly, one of the most efficient and practical frameworks has been developed by Dawid & Skene in 1979 for estimating individual error-rates for patients examined by several clinicians [14]. Sheng, Provost & Ipeirotis adopted the Dawid-Skene algorithm for estimating the worker quality and inferring the correct answers in crowdsourcing applications and provided an implementation [15]. Similarly, Whitehill et al. presented the generative model of labels, abilities, and difficulties (GLAD), which simultaneously infers the correct answers and estimates both worker quality and task difficuly [16]. Demartini, Difallah & Cudré-Mauroux developed the ZenCrowd model, which exploits factor graphs and belief propagation for aggregating the answers received from the workers and blacklisting the unreliable ones [17].

Karger, Oh & Shah presented an order-optimal iterative algorithm for answer aggregation in binary choice crowdsourcing tasks [18]. A highly important finding of their work is the *phase transition* effect, which affects virtually all the answer aggregation methods as follows. Given the number of answers per task l, the number of answers per worker r, and the collective quality of a crowd q, no method is performing better than a simple majority voting heuristic when $(l-1)(r-1)q^2 < 1$.

Implementing an adaptive crowdsourcing method requires expertise both in numerical methods and statistical inference. Hence, there exist software packages like SQUARE created by Sheshadri & Lease [19], which contains implementations of such methods as ZenCrowd [17], naïve Bayes [20], etc.

Another trending topic is online algorithms, which differ from the above mentioned—offline—ones in operating during the data collection process in real time. Welinder & Perona presented an online algorithm for estimating annotator parameters that requires expert annotations to assess the performance of the workers [21]. Fan et al. proposed the iCrowd model, which handles adaptive task allocation, worker ranking and answer aggregation simultaneously [22]. Unfortunately, implementation of the latter is unavailable for general public.

III. Problem

Let S be the set of synsets, which are the sets of quasi-synonyms. Let R be the set of hypernymic/hyponymic relations between the individual words, i.e. the word pairs (g, s), where g is the genus and s is the species of the given pair.

Suppose that one desires to link the S synsets to each other w.r.t. R, i.e. establish the hyponymic/hypernymic relations between them. This problem is especially important in cases when two different lexical resources are getting merged, because it is necessary to keep both resources unambiguous.

A naïve way of doing so is iterating over each $(g, s) \in R$, fetching the set of genus candidate synsets S_g, the set of species candidate synsets S_s, and then manually checking whether the *is-a* relation is present, or not, in each synset pair belonging to $S_g \times S_s$. For instance, a similar experiment on a subset of Yet Another RussNet (YARN) sense inventory [3], in which $|S| \approx 6105$ and $R \approx 29764$, resulted in the number of 30177 human operations, which is unacceptable due to performance reasons, since most of such pairs have no such a relationship. Note that the situation is getting even worse in crowdsourcing applications, when answer redundancy is required due to the unknown expertise of the workers.

In order to reduce the amount of work needed to be done, it seems to be reasonable to split the original problem into simpler ones, solving which could be (1) assisted by a computer, (2) possibly done by several human workers in a parallel way, and (3) require less amount of operations.

IV. Approach

The original problem is getting divided into two separate ones: (1) providing automatically matched data for crowd annotation and (2) the actual annotation. The high-level view of the proposed system is shown at Fig. 1 and the following sections will describe the main components of the system.

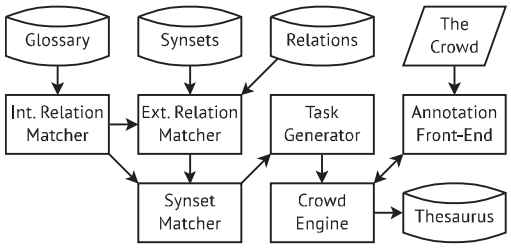

Fig. 1. Architecture of the proposed system

The processing starts with the internal relation matching procedure, which extracts candidate *is-a* relations from the glossary. Since that the domain-specific glossaries include such terms as *reporting system* and *regional reporting system*, it seems to be reasonable to treat these entries as separate candidates. In order to deal with syntactic structure of the multiword terms, a dependency parser has to be used (in fact, any other grammar can be used, but dependency graphs are particularly easy to process and operate). Having constructed the domain-specific set of candidate relations R_{Int}, the set is getting merged with the external resource set of relations R_{Ext}. Finally, after the synset matching procedure, the tasks for crowd workers are getting automatically generated and deployed on a crowdsourced annotation engine.

A. Internal Relation Matching

Algorithm 1 takes the dependency graph (N, A) of a given term, where N is the set of words and A is the set of binary, irreflexive arc relations on N. Since that the algorithm is recursive, the n parameter specifies the root node on first iteration or, otherwise, the current node. The algorithm returns the set of extracted candidate relations R_{Int}. The $relation(\cdot)$ function returns the relation type represented by the argument. The *ROLES* set includes three relations: predicative, agent and quasi-agent, which could be mapped into hypernymic/hyponymic relations.

Algorithm 1 IntRelMatch$(N, A, r = \text{ROOT}) \to R_{Int}$

$R_{Int} \leftarrow \emptyset$
$children \leftarrow \{n : n \in N \wedge \exists(r, n) \in A\}$
for all $n \in children$ **do**
  if $relation(n) \in$ *ROLES* **then**
    $R_{Int} \leftarrow R_{Int} \cup$ IntRelMatch(N, A, n)
  end if
end for
$d \leftarrow \{n : n \in children \wedge relation(n) = \textit{definitive}\}$
for $i = |d|$ **to** 1 **do**
  $R_{Int} \leftarrow R_{Int} \cup (r, d_{i \ldots |d|})$
end for
return distinct R_{Int}

B. External Relation Matching

Algorithm 2 matches the hyponyms of the thesaurus with the hypernyms of the glossary for deriving the universal relation set $R = R_{Int} \cup R_{Ext}$, where R_{Int} set is provided by the Algorithm 1 and R_{Ext} is provided by the thesaurus. Having matched the hypernyms and hyponyms, the algorithm emits pairs for matching the corresponding synsets. The purpose of this algorithm is to reduce the amount of work made by the workers and filter out relations that are not represented by both resources.

C. Synset Matching

Algorithm 3 is inspired by an observation that if a pair of synsets share at least three distinct words, then the synsets are likely to represent the same lexical sense [23]. Exploiting such a phenomenon makes it possible to perform synset deduplication, which results in less amount of work performed by the workers.

Algorithm 2 ExtRelMatch$(R_{Int}, R_{Ext}) \to P$

$P \leftarrow \emptyset$
$v \leftarrow \{s : \exists(g, s) \in R_{Ext}\} \cap \{g : \exists(g, s) \in R_{Int}\}$
for all $s \in v$ **do**
  for all $g \in \{g : \exists(g, s) \in R_{Ext}\}$ **do**
    $P \leftarrow P \cup \{(g, s)\}$
  end for
end for
return distinct P

Algorithm 3 SynsetMatch$(P, S) \to (S_g, S_s)$

$S_g \leftarrow \emptyset$, $S_s \leftarrow \emptyset$
for all $(g, s) \in P$ **do**
  for all distinct $\vec{s} \in \{\vec{s} : \exists \vec{s} \in S \wedge g \in \vec{s}\}$ **do**
    $S_g \leftarrow S_g \cup \{\vec{s}\}$
  end for
  for all distinct $\vec{s} \in \{\vec{s} : \exists \vec{s} \in S \wedge s \in \vec{s}\}$ **do**
    $S_s \leftarrow S_s \cup \{\vec{s}\}$
  end for
end for
return (**distinct** S_g, **distinct** S_s)

D. Genus-Species-Match Workflow

In the Genus-Species-Match workflow, presented at Fig. 2, the annotation is composed of three stages: the first two run independently, the last one runs after the first two have been completed.

- Genus. Given a genus-species pair $(g, s) \in R$ and a synset $\vec{s} \in S_g$, a worker has to confirm whether $\vec{s}$ represents the *genus* of s (Fig. 3).
- Species. Given a genus-species pair $(g, s) \in R$ and a synset $\vec{s} \in S_s$, a worker has to confirm whether $\vec{s}$ represents the *species* of g (Fig. 4).
- Match. Given a pair of synsets $(\vec{s_g}, \vec{s_s})$, a worker has to confirm that this pair represents a reasonable *is-a* relation, or not (Fig. 5).

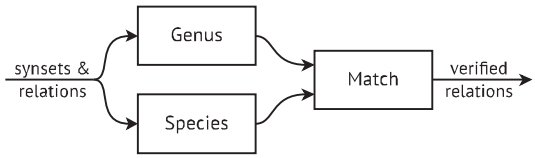

Fig. 2. The proposed Genus-Species-Match workflow

Example: at the *Genus* stage, the worker has to specify whether the synset {advertisement, announcement, advert} is the *genus* of the pair (advertisement, *denim*), or not. Similarly, at the *Species* stage, the worker has to specify whether the synset {jacket, coat} is the *species* of the pair (*clothes*, jacket), or not. Then, at the *Match* stage, the worker has to specify whether the synsets {clothes, dress, apparel} and {coat, jacket} are really connected by the *is-a* relationship. Hence, instead of inconvenient number of operations mentioned in the Section III, the whole annotation process will take approximately $O(|S_g| + |S_s|)$ operations, excluding the "Match" stage, the running time of which is the least.

Is the "$\vec{s}$" concept a genus of the word s?
[ ] Yes
[ ] No

Fig. 3. Task format for the *Genus* stage

Is the "$\vec{s}$" concept a species of the word g?
[ ] Yes
[ ] No

Fig. 4. Task format for the *Species* stage

V. Experiments

The proposed workflow is empirically evaluated as follows. Firstly, the EMERCOM glossary has been processed by Algorithm 1 for extracting candidate relations. MaltParser for Russian has been used as the dependency parser [24], since it is available as a properly built container image [25].

Secondly, the obtained internal relations have been merged with ones from YARN using the Algorithm 2, which resulted in the total number of 383 genus-species pairs. YARN is an open source project aimed at creation of a large electronic thesaurus for Russian through the use of crowdsourcing [3]. The synsets in YARN are created using a collaborative online editor, while the hypernymic/hyponymic relations are imported from a machine-readable version of the Russian Wiktionary [26].

Finally, 1438 synsets for the *Genus* stage and 833 synsets for the *Species* stage have been produced by Algorithm 3 from these pairs. In this study, two synsets are considered as duplicates if they share at least two words.

A. Any Marketplace Here?

In order to invite real crowd workers to the experiment, several online labor marketplaces, including MTurk [27], CrowdFlower [28], microWorkers [29], Yandex.Toloka [30] and TurboText [31], have been studied w.r.t. the *median* reward per HIT, the number of submitted distinct HIT types, and availability to requesters and workers from Russia (Table I). The observed difference in prices is caused by the possibility to group several atomic tasks into batches on MTurk and Yandex.Toloka. It is interesting that the latter offers larger work amounts for a significantly lower reward.

TABLE I. Online labor marketplaces

	$ per HIT	# of HITs	Requesters	Workers
MTurk	0.60	3050	No	No
CrowdFlower	0.05	47	Yes	Yes
microWorkers	0.65	93	Yes	Yes
Yandex.Toloka	0.01	3	No	Yes
TurboText	0.06	289	Yes	Yes

Having conducted the market study, TurboText has been chosen as the online labor marketplace for the experiment primarily for the reason of it also being a copywriting marketplace, which implies presence of skilled Russian native speakers.

As for October 1, 2015, the exchange rate set by the Bank of Russia is 65.73 RUR per \$1. The *average* reward on TurboText is 11 RUR per microtask, which is approximately \$0.17. The price per microtask has been set to 5 RUR (Russian rubles), so a worker is getting rewarded by approximately \$0.08 per task.

Is the concept "$\vec{s_s}$" a species of the concept "$\vec{s_g}$"?
[ ] Yes
[ ] No

Fig. 5. Task format for the *Match* stage

B. Annotation Setup

The annotation process has been running on Mechanical Tsar, an open source crowdsourcing engine, designed for rapid deployment of adaptive mechanized labor workflows [32].

A screenshot of the worker interface at Fig. 6 shows example of the task "Is the concept $\{\text{person}, \text{client}, \text{partner}\}$ a species of the word *party*?" belonging to the stage "Species". For convenience of the workers, each word hyperlink opens a separate tab with a Google search page corresponding to the exact word match.

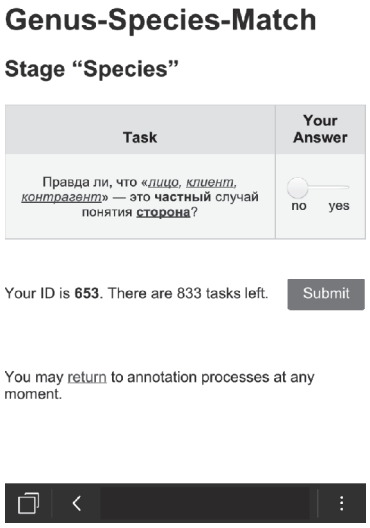

Fig. 6. Example of a worker interface (captions are translated into English for convenience of the readers, while the task description is intentionally left in Russian)

In order to provide the workers with better annotation experience, the tasks were grouped in batches, i.e. one microtask represents one batch containing several tasks. Each task is annotated at least five times by distinct workers. When displaying synsets, only five top frequency words as according to the Russian frequency dictionary have been shown [33]. Actual annotation setup is present at Table II. The majority voting heuristic is used for aggregating the answers received at the stages "Genus" and "Species", resulting in 287 tasks for the "Match" stage.

TABLE II. Annotation setup

	# of answers	# in batch	RUR per batch
Genus	5	15	5.00
Species	5	15	5.00
Match	5	12	5.00

Since all the tasks in the Genus-Species-Match workflow assume binary choices, it is reasonable to adopt the best

practices developed in similar studies. Snow et al. showed that in binary choice tasks it is sufficient to obtain at least 4 answers per task from the crowd workers to get a reliable output [20]. However, this case is prone to ties, when two workers voted for the first answer and two other workers voted for another one. As it has also been discussed in the Section II-C, Karger, Oh & Shah recommend using at least 5 answers per task for exploiting sophisticated statistical answer aggregation techniques, which outperform majority voting [18]. To study how an answer aggregator affects the result quality, the "Match" stage answers have been processed by three different aggregators: the above mentioned majority voting model, an iterative algorithm by Karger, Oh & Shah (KOS) [18], and the ZenCrowd model [17]. It should be noted that original variants of the two former methods are proprietary, hence, an open source implementation of ZenCrowd has been integrated from SQUARE [19], while implementation of KOS has been already provided by Mechanical Tsar [34].

Additionally, to make sure that crowdsourcing such a task is a reasonable kind of activity, a naïve automatic approach has been compared with the crowd results. Normally, while generating the "Match" tasks, the system processes tuples of $(\vec{s_g}, \vec{s_s} | g, s)$ per each genus-species pair (g, s), where $\vec{s_g} \in S_g$ is the synset obtained from the "Genus" stage, $\vec{s_s} \in S_s$ is the synset obtained from the "Species" stage. In order to set up the baseline to compare with, a pair of synsets is getting automatically *matched* if they share at least two words in the external resource set of relations, i.e.

$$\left|\{s' : \exists(g, s') \in R_{Ext}\} \cap \{g' : \exists(g', s) \in R_{Ext}\}\right| > 1.$$

VI. Results & Discussion

The whole annotation process took 264 minutes, i.e. $max(206, 197) = 206$ minutes for the simultaneous *Genus* and *Species* stages, correspondingly, and 58 minutes for the consequent *Match* stage. As the result, 47 distinct workers have submitted 13056 answers (see Table III and Fig. 7). Completely sequential execution will take at least 461 minutes.

TABLE III. Annotation summary

	Workers	Tasks	Answers	Agreement
Genus	39	1438	7357	59%
Species	18	833	4205	59%
Match	18	287	1494	55%
Total	47	2558	13056	

For verification purposes, an expert annotator has assessed the final output. Then, given the number of true positives TP, true negatives TN, false positives FP and false negatives FN (Table IV), the precision $P = \frac{TP}{TP+FP}$, recall $R = \frac{TP}{TP+FN}$ and $F_1 = 2\frac{P \cdot R}{P+R}$ have been computed [35].

TABLE IV. Experimental results

	TP	TN	FP	FN	P	R	F_1
Baseline	40	102	17	128	0.70	0.24	0.36
Maj. Voting	129	57	62	39	0.68	0.77	0.72
KOS	142	63	56	26	0.72	0.84	0.78
ZenCrowd	146	69	50	22	**0.74**	**0.87**	**0.80**

According to the experimental results, the ZenCrowd model outperformed all other algorithms in terms of precision, recall and F_1 score at the same time (Table IV).

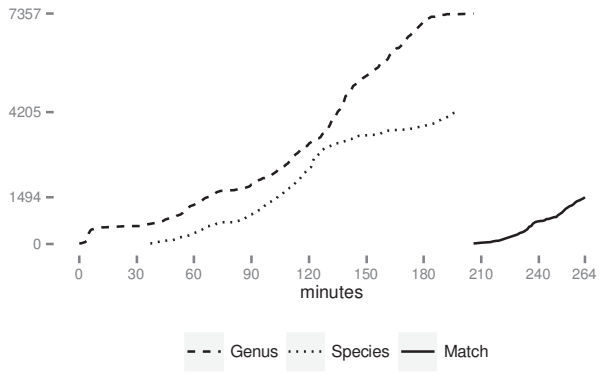

Fig. 7. Cumulative number of received answers by process over time

A. Errors & Agreement

After a thorough analysis of the observed errors made while aggregating the answers by ZenCrowd, three notable mistake types have been found:

a) worker issues: the workers made mistakes by incorrectly interpreting the lexical sense,

b) synset issues: the synsets mapped to the relations have either too broad or too narrow meaning,

c) data issues: the matched synsets or relations inherit nonsense from the YARN data, like *city* as the genus in the genus-species pair $(\text{region}, \text{city})$ or such odd synsets as $\{\text{country}, \text{capital}, \text{province}, \text{district}\}$, etc.

Despite that each task has been processed by at least five distinct workers, the crowd agreement is relatively low (Table III). It seems that the workers were confused primarily by the data issues. Although the Genus-Species-Match workflow performed robust, the error rate could be even lowered by providing it with less noisy input data.

B. Other Labor Marketplaces

TurboText, mentioned in the Section V-A, has been selected as the online labor marketplace since it has a wide audience, while having a microtask support. However, it was highly interesting to briefly evaluate the difference between a microtask marketplace and a traditional one. Thus, the following study has been conducted on a well-known online labor marketplace in Russia—it will be denoted as *X*.

Simultaneously with submission of microtasks to TurboText, a regular task on *X* has been created for providing answers for each task of the *Species* stage. Only two potential workers contacted the requester. The first one was a bot, who replied within a few seconds right after the posting specifying no details about the work. The second one was a real person, who asked for more than $\frac{1}{2}$ of the overall budget spent on TurboText. The task has been removed from *X* after the completion of the *Genus* and *Species* stages on TurboText. However, it seems to be highly topical to compare microworkers with regular ones w.r.t. the annotation quality.

VII. Conclusion

Genus-Species-Match involves several tightly coupled data processing mechanisms (Fig. 1) for both disambiguating simultaneously the synsets and establishing hyponymic/hypernymic relations between them.

As it has been mentioned in the Section I, the system was initially designed to efficiently insert a glossary into a sense inventory, but it can definitely be used just for synset disambiguation by omitting the internal relation matching procedure and replacing the external relation matching procedure with an arbitrary one.

We see several directions for future work:

- using a similar system for establishing holonymy/meronymy relations,
- developing a budget allocation algorithm for Genus-Species-Match, like Budgeteer for the well-known Find-Fix-Verify workflow [36],
- improving the preprocessing pipeline with various sophisticated matching techniques, including ones from distributional semantics [37].

In previous studies, we noted that mechanized labor crowdsourcing workflows may barely be exploited in Russia due to the limited amount of available crowd workers [3], [13], [23], [32], [37]. This has been, fortunately, proven wrong. To the best of our knowledge, this is the first attempt to recruit paid crowd workers from an online labor marketplace for annotating a language resource made in Russia. It opens new possibilities for creating, enhancing and evaluating language resources, knowledge bases, corpora and other useful resources for processing Russian.

The deliverables of this study, including the source code of matchers and generators under the MIT license, as well as the input, intermediate and output data under the CC BY-SA 3.0 license, are available for download on http://ustalov.imm.uran.ru/pub/gsm-ainl.tar.gz.

Acknowledgment

The author has no affiliation with online labor marketplaces mentioned in this paper. This work is supported by the Russian Foundation for the Humanities, project no. 13-04-12020 “New Open Electronic Thesaurus for Russian”. The author would like to thank all those who participated in the described experiment. He is also grateful to the anonymous referees who offered very useful comments on the present paper.

References

[1] I. Gurevych and J. Kim, *The People’s Web Meets NLP*, Springer Berlin Heidelberg: Berlin, 2013.

[2] Glossary of EMERCOM, Web: http://www.mchs.gov.ru/dop/terms.

[3] P. Braslavski, D. Ustalov, M. Mukhin, “A Spinning Wheel for YARN: User Interface for a Crowdsourced Thesaurus”, *in Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, Apr. 2014, pp. 101-104.

[4] N.A. Astrakhantsev, D.Y. Turdakov, “Automatic construction and enrichment of informal ontologies: A survey”, *Programming and Computer Software*, vol.39(1), Feb. 2013, pp. 34-42.

[5] M.A. Hearst, “Automatic Acquisition of Hyponyms from Large Text Corpora”, *in Proceedings of the 14th Conference on Computational Linguistics*, vol.2, Aug. 1992, pp. 539-545.

[6] D. Yang, D.M. Powers, “Automatic Thesaurus Construction”, *in Proceedings of the Thirty-first Australasian Conference on Computer Science*, vol.74, Jan. 2008, pp. 147-156.

[7] C. Giuliano, A.M. Gliozzo, A. Gangemi, K. Tymoshenko, “Acquiring Thesauri from Wikis by Exploiting Domain Models and Lexical Substitution”, *The Semantic Web: Research and Applications*, vol.6089, May 2010, pp. 121-135.

[8] M. Hagiwara, Y. Ogawa, K. Toyama, “PLSI Utilization for Automatic Thesaurus Construction”, *Natural Language Processing — IJCNLP 2005*, vol.8876, Oct. 2005, pp. 334-345.

[9] C. Sarasua, E. Simperl, N.F. Noy, “CrowdMap: Crowdsourcing Ontology Alignment with Microtasks”, *in The Semantic Web – ISWC 2012*, vol.7649, Nov. 2012, pp. 525-541.

[10] M.S. Bernstein, G. Little, R.C. Miller, B. Hartmann, M.S. Ackerman, D.R. Karger, D. Crowell, K. Panovich, “Soylent: A word processor with a crowd inside”, *in Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*, Oct. 2010, pp. 313-322.

[11] J. Wang, T. Kraska, M.J. Franklin, J. Feng, “CrowdER: Crowdsourcing Entity Resolution”, *Proc. VLDB Endow.*, vol.5(11), Jul. 2012, pp. 1483-1494.

[12] C. Biemann, “Creating a system for lexical substitutions from scratch using crowdsourcing”, *in Language Resources and Evaluation*, vol.47(1), Mar. 2012, pp. 97-122.

[13] D. Ustalov, Y. Kiselev, “Add-Remove-Confirm: Crowdsourcing Synset Cleansing”, *in 2015 IEEE 9th International Conference on Application of Information and Communication Technologies (AICT)*, Oct. 2015, pp. 143-147.

[14] A.P. Dawid, A.M. Skene, “Maximum Likelihood Estimation of Observer Error-Rates Using the EM Algorithm”, *Journal of the Royal Statistical Society*, vol. 28(1), 1979, pp. 20-28.

[15] V.S. Sheng, F. Provost, P.G. Ipeirotis, “Get Another Label? Improving Data Quality and Data Mining Using Multiple, Noisy Labelers”, *in Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Jun. 2008, pp. 614-622.

[16] J. Whitehill, P. Ruvolo, T. Wu, J. Bergsma, J. Movellan, “Whose Vote Should Count More: Optimal Integration of Labels from Labelers of Unknown Expertise”, *in Advances in Neural Information Processing Systems 22*, Dec. 2009, pp. 2035-2043.

[17] G. Demartini, D.E. Difallah, P. Cudré-Mauroux, “Large-scale Linked Data Integration Using Probabilistic Reasoning and Crowdsourcing”, *The VLDB Journal*, vol.22(5), Jul. 2013, pp. 665-687.

[18] D.R. Karger, S. Oh, D. Shah, “Budget-Optimal Task Allocation for Reliable Crowdsourcing Systems”, *Operations Research*, vol.62(1), Feb. 2014, pp. 1-24.

[19] A. Sheshadri, M. Lease, “SQUARE: A Benchmark for Research on Computing Crowd Consensus”, *in First AAAI Conference on Human Computation and Crowdsourcing*, Nov. 2013, pp. 156-164.

[20] R. Snow, B. O’Connor, D. Jurafsky, A.Y. Ng, “Cheap and Fast—but is It Good?: Evaluating Non-expert Annotations for Natural Language Tasks”, *in Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Oct. 2008, pp. 254-263.

[21] P. Welinder, P. Perona, “Online crowdsourcing: Rating annotators and obtaining cost-effective labels”, *in 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, Jun. 2010, pp. 25-32.

[22] J. Fan, G. Li, B.C. Ooi, K-l Tan, J. Feng, “iCrowd: An Adaptive Crowdsourcing Framework”, *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, Jun. 2015, pp. 1015-1030.

[23] Y. Kiselev, D. Ustalov, S. Porshnev, “Eliminating fuzzy duplicates in crowdsourced lexical resources”, Proc. of the Eight Global Wordnet Conference, in press.

[24] S. Sharov, J. Nivre, “The proper place of men and machines in language technology: Processing Russian without any linguistic knowledge”, *in Computational Linguistics and Intellectual Technologies: papers from the Annual conference “Dialogue”*, vol.10(17), May 2011, pp. 657-670.

[25] Docker Hub, Web: https://hub.docker.com/r/nlpub/maltparser/.

[26] A.A. Krizhanovsky, A.V. Smirnov, “An approach to automated construction of a general-purpose lexical ontology based on Wiktionary”, *Journal of Computer and Systems Sciences International*, vol.52(2), Mar. 2013, pp. 215-225.

[27] Amazon Mechanical Turk - Welcome, Web: https://www.mturk.com/mturk/welcome.

[28] CrowdFlower | People-powered Data Enrichment Platform, Web: https://www.crowdflower.com/.

[29] Microworkers - work & earn or offer a micro job, Web: https://microworkers.com/.

[30] Yandex.Toloka, Web: https://toloka.yandex.com/.

[31] TurboText, a convenient copywriting market, Web: http://www.turbotext.ru/.

[32] D.A. Ustalov, “A Crowdsourcing Engine for Mechanized Labor”, *Proceedings of the Institute for System Programming*, vol.27(3), Jul. 2015, pp. 351-364.

[33] O.N. Lyashevskaya and S.A. Sharov, *The frequency dictionary of modern Russian language*, Moscow: Azbukovnik, 2009.

[34] Mechanical Tsar, Web: http://mtsar.nlpub.org/.

[35] D.M.W. Powers, “Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation”, *Journal of Machine Learning Technologies*, vol.2(1), Feb. 2011, pp. 37-63.

[36] L. Tran-Thanh, T.D. Huynh, A. Rosenfeld, S.D. Ramchurn, N.R. Jennings, “Crowdsourcing Complex Workflows under Budget Constraints”, *in Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, Jan. 2015, pp. 1298-1304.

[37] A. Panchenko, N.V. Loukachevitch, D. Ustalov, D. Paperno, C.M. Meyer, N. Konstantinova, “RUSSE: The First Workshop on Russian Semantic Similarity”, *in Computational Linguistics and Intellectual Technologies: papers from the Annual conference “Dialogue”*, vol.2, May 2015, pp. 89-105.