Exact-match Based Wikipedia-WordNet Integration

Tomasz Boiński
Faculty of Electronics, Telecommunication and Informatics
Gdańsk University of Technology
Gdańsk, Poland
tobo@eti.pg.edu.pl

Julian Szymański
Faculty of Electronics, Telecommunication and Informatics
Gdańsk University of Technology
Gdańsk, Poland
julian.szymanski@eti.pg.edu.pl

Tymoteusz Cejrowski
Faculty of Electronics, Telecommunication and Informatics
Gdańsk University of Technology
Gdańsk, Poland
tymoteusz.cejrowski@pg.edu.pl

***Abstract*—Ability to link between WordNet synsets and Wikipedia articles allows usage of those resources by computers during natural language processing. A lot of work was done in this field, h owever m ost o f t he a pproaches f ocus o n similarity between Wikipedia articles and WordNet synsets rather than creation of perfect matches. In this paper we proposed a set of methods for automatic perfect matching generation. The proposed methods were evaluated and integrated into one unified solution for generating matches with good quality. The paper describes and evaluates the proposed methods and presents the integration process. The evaluation of the final proposed solution is given.**

***Index Terms*—Wikipedia, WordNet, mapping, exact-match**

I. Introduction

Wikipedia and other similar resources are a vast information source. Such data is, however, stored in a form readable for humans, as they are the prime users of the system. More often there is a need of automatic data processing required e.g. in information lookup or question answering. For that purpose the data needs to be formalized.

The formalization process began with establishing of the Semantic Web initiative [1]. It aimed at extending the Web with meta data [2], [3]. The data is enriched with links to external structuralised repositories like ontologies [4] and thus introducing semantic connections between the concepts in the unstructured text. This approach is strictly tied with the language that the data creators use and needs to be promptly updated with cultural and linguistic changes requiring some automation of their construction. As such only domain-oriented ontologies are usually used, e.g. [5], [6].

Due to the problems with high formalization of ontologies different approaches for data enriching were introduced. The most popular are Semantic Networks [7] based on typed relations between concepts representing word meanings. Unfortunately there is a lack of large-scale and high-quality Semantic Networks which in turn is the main obstruction in the development of natural language technologies.

One of the newest trend in structuring data is so called Linked Data [8]. In this approach the data from different knowledge sets are connected with links. The links creation approaches range from the strict logic-based methodologies that are using Ontologies [9] to less formal Folksonomies [10].

The approach allows usage of unstructured data based on the formal structure of the other data set making the data as a whole more useful for computers. When linking the resources, we can provide a model of knowledge that is based on typed references [11]. In this approach the main problem is the link creation itself. Linking the resources by hand is not feasible as this is a lengthy process requiring a lot of human resources. On the other hand automatically created links usually don't have high quality and require verification. The links quality is in this case limited by low quality of natural language processing methods. Some progress have been however made in this field.

In this paper we present an automated approach for exact linking of data between Wikipedia and WordNet. The links created are perfect matches, meaning that the Wikipedia article either is describing the same concept as a WordNet synset or the match is not created.

The structure of the paper is as follows. Section II presents existing solutions used for Wikipedia-WordNet mapping. Section III presents the proposed algorithms and describes a reference set used during the evaluation. Next, the section IV shows aggregated quality assessment of the proposed algorithms and their variants, and than section V presents evaluation of the proposed algorithms and method for selecting and ordering the final set of approaches to perfect matching between Wikipedia articles and WordNet synsets. Finally some conclusions are given.

II. Existing Solutions

One of the first approaches to link generation was done by Ruiza-Casado et al. [12]. In this approach WordNet was mapped to Simple English Wikipedia. In this approach the Wikipedia page was first cleaned up (all formatting was removed, from the text only nouns, verbs, adjectives and adverbs were taken into account). The title of the page was than mapped to WordNet synsets. If there exists one synset containing the article title it was set as perfect match, if there was none the team assumed that there is no synset matching the article. If there was more than one synset matching, the article title was checked against the synset description. Using this approach the authors obtained high precision - over 91% of the articles were mapped. Similar approach was taken by [13]. In this case additional information in the title, given in brackets, was also used. In this case the precision achieved was over 89% with 89.5% of coverage. At time of the work the Simplified English Wikipedia contained however only 4100 articles.

In work [14] Wikipedia articles were not connected with synsets but with senses. The algorithm used was based on context disambiguation for senses and articles. In other words a set of words was created that was related to both the Wikipedia article and WordNet synset. For the disambiguation context $Ctxpp(s)$ of sense s of synset S we consider: synonyms of S, synonyms of hypernyms and homonyms of S, synonyms of synsets that share direct hypernym with S and words in definition of S. For Wikipedia articles w the disambiguation context $Ctx(w)$ is defined by words in title refinements (in brackets), title of Wikipedia pages that are linked from the article w and category names of the article w. For all monosemic both in WordNet and Wikipedia (meaning for given article title only one Wikipedia page and one WordNet synset is found) the page is connected with the synset. For articles that are redirects for which we already have a connection we also connect the article with the synset that the redirect page connected to. In other cases the article is connected with a sens s for which probability $p(s|w)$ is the highest, where s belongs to senses of article w. The authors claim that the method gives good results with precision around 82% and coverage over 77%.

In [15] authors proposed an algorithm that allowed matching more than one Wikipedia article to a single synset. In this case the algorithm first creates a set of probable candidates for the given synset. The set of candidates is build similarly as in [14]. In the next step the articles are checked to select those which match the synset in terms of meaning. This is done either by checking cosine distance between the synset and the article or by using PageRank algorithm [16] modified by the authors to better match the WordNet structure [17]. The method provided good results with F1-measure equal to 0.78%.

In [18]–[20] we proposed 4 algorithms based mainly on synset synonyms and article titles. The candidate articles are obtained using Opensearch API [21]. The first algorithm connects a synset to an article only when as a result of Wikipedia search based on the one of the synset synonyms we obtain only one article. Test done on 100 article shows that this approach is characterized by high precision (around 97%). The second algorithm matches an article when it shows on results lists of searches done using at least 2 synonyms. This approach itself has accuracy around 88%. The third algorithm is based on the observation that most of the synsets has only one synonym. In such case, when the synonym is identical to a Wikipedia article title the match is created. Precision of this approach is around 88%. The last algorithm creates a match between a synset and an article that was returned as first on Opensearch API result list. This approach found matches for 84% of the articles, however only 17% of them were correct. The combination of the four algorithms was also checked. When the first three were used F-measure was equal 0.7 with precision 73% and recall 68%. We further improved the solution using crowdsourcing approach [18], [22], [23]. For this we employed so called Games with a Purpose [24].

The aforementioned approaches focused on providing a best effort connection without considering whether the match is perfect or matches just similar concepts. In our current work we focused on creation of links that were exact matches.

III. Exact Match Finding Algorithms and their Evaluation

In this section we present the proposed algorithms and their the most important variants, denoted as MX_Y - Y variant of algorithm X. In the further sections the algorithms will be referenced using this denotation.

A. Reference Set and Wikipedia Articles Lookup

For the preliminary evaluation of the algorithms we created a reference set RS consisting of randomly selected 200 synsets S with N synonyms s_i. The synsets were denoted as $< s_1,\ s_2,\ ...,\ s_N >$. For 132 synsets S we manually assigned a set of Wikipedia articles $A_C(S)$. Remaining 68 synsets did not have matching articles. The algorithms were than tested for the selected 200 synsets S.

As comparing the synset with the whole Wikipedia is not feasible (as there currently are 5,854,295 articles in English Wikipedia alone [25]) we limited the number of article pages for which each synset was validated. The further validation was also done automatically. In all cases the Wikipedia lookup is done automatically using MediaWiki Action API Opensearch [26] feature. In our work we performed the query for each synonym s_i of S. The results were than aggregated as set $A(C)$.

The important factor is the number of articles returned for each synonym, as there is a difference in synsets synonyms and articles naming. WordNet uses only characters from English alphabet whereas Wikipedia allows any characters. For example a WordNet synset $< heloise >$ refers to a person whose real name is "Héloïse". The lookup result differs depending on what characters where used - the article is returned as first when using the proper case and as seventh when using only English characters. The synsets and the articles can also have different naming convention used thus resulting in moving the proper article page up and down on the results list. Our tests showed that setting the limit to 20 makes the best compromise between the number of pages returned, probability of the direct match page being returned and the performance of the lookup.

B. Article Title and Synsets Identity

The first proposed algorithm is the simplest one. We assume that if for article a the title $title(a)$ matches at least one of the synset S synonyms s_i the article matches the synset. For the reference set RS we found 104 articles for which $title(s) \in s(S)$, namely the title of the algorithm matches at least one of the synonyms of given synset in WordNet. In 76 cases the article found was a perfect match. For further 14 articles one of the articles on the result list was a perfect match (algorithm M1_1).

This approach works very well for uncommon names like Latin names of species. In other cases it also works well, however it is prone to errors if at least one of the synsets produces a lot of results e.g. for synset $< ham,\ ham\ actor >$ we can obtain misleading results. There is also a problem of case sensitivity. WordNet synsets synonyms are always spelled using lowercase, Wikipedia article pages are always started with capital letter and written according to the grammar. This can lead to problems as e.g. synset $< seat >$ will be matched to both article $Seat$ and $SEAT$, both having completely different meaning.

This approach needs some special attention when handling redirects in Wikipedia as in some cases there are synsets that has synonyms that in Wikipedia are only redirects to a proper page, e.g. $< pobeda\ peak,\ pobedy\ peak >$ which redirects to $Jengish\ Chokusu$ article and in WordNet there is no synset with a synonym $jengish\ chokusu$. The procedure goes as follows, where by $redirects(a)$ we denote a set containing the article and all pages redirecting to it (algorithm M1_2):

1) if there is only one pair a, s such that $s \in redirects(a)$ map the article on the synset;
2) if there are more than one pair a, s such that $s \in redirects(a)$:
    a) if set $redirects(a_n)$ contains more synonyms of S than other for other articles a_i than the article a_n on the synset;
    b) in other case do not create a mapping.

Using such procedure the algorithm created mappings for 133 synsets (from the reference set) out of which 93 were correct. When we extended the conditions that at least 2 synonyms have to match the redirect pages the procedure created 37 mappings (for synsets from the reference set) out of which 32 were correct (algorithm M2_3).

The last case requires further investigation as in many cases Wikipedia article title has additional words attached, e.g. Wikipedia contains an article titled $Yam\ (vegetable)$. In the first case it will not match to any WordNet synonym due to the $(vegetable)$ suffix. In such case we split the title t into two parts, the first being the main title without the added words in the brackets, denoted $T1(t)$ and the one containing the suffix, denoted $T2(t)$. We thus create a match if the synsets S synonym matches $T1(t)$ and $T2(t)$ is not empty and $T2(t) \in glossa(S)$, where $glossa(S)$ is the synset S definition (algorithm M2_4).

C. Uniqueness of the Article on the Results List

The second algorithm is based on the assumption that if for any of the synset synonyms there is only one article returned we assume that this is the direct match. Such situation is expected to occur especially for synonyms with unique or long names.

In the reference set RS the proposed algorithm found 46 synsets for which there was exactly one synonym that had exactly one article as a query result. This way the proposed algorithm correctly assigned an article to 30 of them (65.22% precision, algorithm M2_1). In 14 cases we found more than one synset with exactly one query result. In all cases the synonyms led to the same article. In 13 cases the resulting articles were a correct match. In the remaining case it can be treated as correct as the algorithm assigned an article $Permanent\ teeth$ to synset $< adult\ tooth,\ permanent\ tooth >$ (algorithm M2_2).

In 4 additional cases there were more than one synonym leading to exactly one article. The target article was however different for each of the synonyms. In such case we assigned an article to a synset if the article was a sole query result for the biggest number of synonyms. In all 4 cases the match created by the algorithm was correct (algorithm M2_3).

D. Similarity Between the Synset Definition and Article Content

The third method bases on the observation that the synset definition in WordNet and article summary in Wikipedia serve the same purpose - to briefly describe the concept. Thus we assume that if they match than the synset and the article also matches.

To calculate the similarity we use cosine similarity between normalized inverse document frequency vectors (eq. 1) of the aforementioned fragments. The vectors v_x are composed of inverse document frequency values idf_i for each word i in the synset definition and article summary.

$$sim(v1, v2) = \frac{v1 \circ v2}{|v1||v2|} \qquad (1)$$

The idf_i is calculated as in eq. 2 where N is the number of documents and n_i is the number of documents containing the word i.

$$idf_i = log\frac{N+1}{n_i+1} + 1 \qquad (2)$$

As the vectors tend to be very long both the synset descriptions and article summaries has to be pre-processed. This includes: lemmatization, stemming, case normalization, stop-list words removal and removal of diacritics characters. As a result the minimal similarity equals 0, maximum is equal 1. If one of the vectors has no elements than the similarity if also equal 0.

The precision and coverage for different values of similarity are presented in Table I (algorithm M3_XX. where XX is the similarity threshold).

We also calculated the precision and coverage relation when the similarity was counted only subset of reference set containing synsets with available mappings and if the vector representation of the synset was nonzero (denoted

TABLE I
PRECISION AND COVERAGE IN RELATION TO SIMILARITY LEVEL

$sim(a, S) >$	Precision (%)	Coverage (%)
0	81	59.88
0.2	61.5	65.85
0.4	15.5	77.42
0.5	5.5	90.91
0.56	1.5	100

TABLE II
PRECISION AND COVERAGE IN RELATION TO SIMILARITY LEVEL FOR RS_{no0}/RS_{sim0}

$sim_{no0}(a, S) >$	Precision (%)	Coverage (%)
0	81	61.73
0.15	78	64.10
0.30	59	72.88
0.45	20.5	75.61
0.50	13.5	74.07
0.55	6	91.67
0.59	4.5	100

as RS_{no0}/RS_{sim0}). The results are presented in Table II (algorithm M4_XX. where XX is the similarity threshold).

In this approach similarity level is critical when deciding whether we should match a synset and an article or not. The final selection will be done during the algorithm evaluation and selection described in the next sections.

E. Algorithm Quality Summary

This section, in Tab. III, aggregates the results described in previous sections for increased visibility. The same algorithms, thanks to their negative conditions, could be also used in a way that allows us to state that a synset does not have a match in Wikipedia articles. The aggregation of all such methods can be found in Tab. IV. In this table RS_0 is the subset of RS containing synsets such that $A(S) \in \varnothing$. Once again the algorithms were labeled for further reference.

TABLE III
PRECISION AND COVERAGE FOR ALGORITHMS WHEN A MATCH IS FOUND

Label	Precision (%)	Coverage (%)
M1_1	52	73.08
M1_2	66.5	69.92
M1_3	18.5	86.49
M1_4	2	75
M2_1	23	65.22
M2_2	7	92.86
M2_3	2	100
M3_00	81	59.88
M3_20	61.5	65.85
M3_40	15.5	77.42
M3_50	5.5	90.91
M3_56	1.5	100
M4_00	81	61.73
M4_15	78	64.10
M4_30	59	72.88
M4_45	20.5	75.61
M4_50	13.5	74.07
M4_55	6	91.67
M4_59	4.5	100

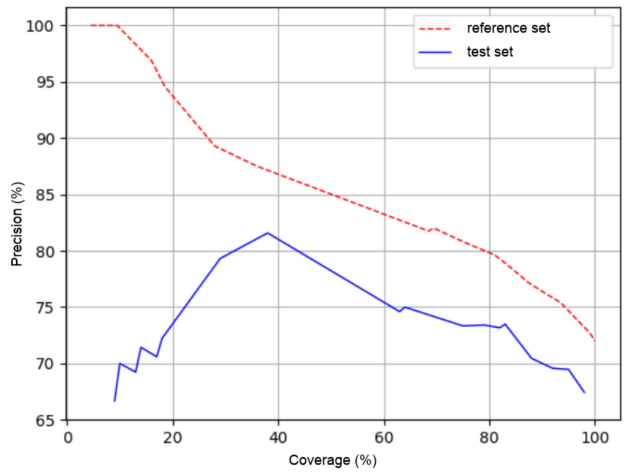

Fig. 1. Precision and coverage ratio for reference set and test set

IV. METHODS AGGREGATION

The methods described in the previous section often overlap or work under certain conditions. To determine the proper order of methods execution we checked how they behave when executed only for synsets for which other methods did not find a match. A fragment of obtained results is presented in Tab. V.

As we can see if the methods are both designed to either create a match or prove that there is no match than the quality of the second method is lowered when run on the failure set of the first method. For increased accuracy it is thus crucial to select a subset of the methods and execute them in the proper order.

The first approach was a greedy one - we manually selected methods based on their precision. in case two methods had the same precision we selected the one with higher coverage. We stopped when reached 100% coverage. This way 24 out of 28 methods were selected and ordered. For the reference set the total combined quality determined by F1-measure was 0.71. When applied to other test set the results were however worse. We thus modified the selection process.

During the second approach we selected methods based on precision calculated for the set of results that were not already matched by currently selected set of methods. Once again we started with a method with the highest precision and the highest coverage. This way we selected 20 algorithms: M4_59, M2_3, B3_06, B2_3, M3_56, M2_2, B1_2, B1_3, M1_3, M1_1, M1_4, B1_1, M3_40, B2_2, B3_10, M2_1, B3_15, B2_1, M3_20, M1_2. The combined F1-measure in this case was 0.72. This approach generated a shorter list of algorithms that gave higher quality.

V. FINAL EVALUATION

For the final evaluation we randomly selected 100 synsets and run the selected algorithms using this set. The precision and coverage ration is presented in Fig. 1.

As we can see the precision of the selected set of algorithm is lower than for the reference set but starting from 40% cover-

TABLE IV
PRECISION AND COVERAGE FOR ALGORITHMS STATING THAT THERE IS NO MATCH

Label	Description	Precision (%)	Coverage (%)
B1_1	$S \in RS_0$	8.5	70.59
B1_2	$S \in RS_0, s(S) > 1$	2.5	80
B2_1	$\neg\exists(s, a) : s \in redirects(a), A(S) > 0$	13.5	48.15
B2_2	$\neg\exists(s, a) : s \in redirects(a), A(S) > 0, s(S) > 1$	3	83.33
B2_3	$\neg\exists(s, a) : s \in T1(t) : t \in redirects(a)$	1	100
B3_06	$\max sim(a, S) < 0.06, S \in RS_{no)}/RS_{sim0}$	0.5	100
B3_10	$\max sim(a, S) < 0.10, S \in RS_{no)}/RS_{sim0}$	3.5	57.14
B3_15	$\max sim(a, S) < 0.15, S \in RS_{no)}/RS_{sim0}$	11.5	39.13
B4_1	$S \in RS_{sim0}$	10.5	76.19

TABLE V
PRECISION AND COVERAGE FOR ALGORITHM PAIRS (C_X - COVERAGE OF ALG. X, P_X - PRECISION OF ALG. X, S_{AB} - SYNSETS MATCHED BY A AND B, $C_{X/Y}$ - COVERAGE OF ALG. X ON SET THAT WAS NOT MATCHED BY ALG. Y), $P_{X/Y}$ - PRECISION OF ALG. X ON SET THAT WAS NOT MATCHED BY ALG. Y)

Alg. A	C_A (%)	P_A (%)	Alg. B	C_B (%)	P_B (%)	S_{AB} (%)	$C_{A/B}$ (%)	$P_{A/B}$ (%)	$C_{B/A}$ (%)	$P_{B/A}$ (%)
M1_1	52	73.08	M1_2	66.5	69.92	48	4	12.5	18.5	51.35
M1_1	52	73.08	M2_1	23	65.22	11	41	70.73	12	45.83
M2_1	23	65.22	M4_00	81	61.73	18.5	4.5	11.11	62.5	56.8
M3_40	15.5	77.42	M4_50	13.5	74.07	12	3.5	71.43	1.5	33.33
B2_1	13.5	48.15	B3_10	3.5	57.14	0.5	13	46.15	3	50
M1_1	52	73.08	B3_15	11.5	39.13	7.5	44.5	76.4	4	37.5
M1_1	52	73.08	B4_1	10.5	76.19	5	47	77.66	5.5	81.82

age the difference is only around 5%. We find this acceptable. For the reference set the highest combined F1-measure equal 0.73 with coverage equal to 93.5% and precision equal to 75.4%. The remaining two algorithms introduced more errors and only applied to very limited number of cases thus were eliminated. The same situation applies for the test set where the 18 algorithms achieved 95% coverage, 69.49% precision and F1-measure equal to 0.68. As a result we propose using the ordered set of 18 algorithms, namely M4_59, M2_3, B3_06, B2_3, M3_56, M2_2, B1_2, B1_3, M1_3, M1_1, M1_4, B1_1, M3_40, B2_2, B3_10, M2_1, B3_15 and B2_1.

VI. CONCLUSION

Ability to link between WordNet synsets and Wikipedia articles allows usage of those resources by computers during natural language processing. A lot of work was done in this field, however most of the approaches focus on similarity between Wikipedia articles and WordNet synsets rather than creation of perfect matches.

In this paper we propose a set of methods for automatic perfect matching generation. The methods were evaluated. Based of their assessment we propose a final selection of the best methods and their ordering as a means to generate matches with good quality.

In the future we plan on extending the evaluation of the proposed methods to further improve their quality. The works needs to be also done on similarity measurement between WordNet synset definitions and Wikipedia articles summaries which might improve the whole process. Furthermore the process of algorithm selection and ordering might be improved to better align their strengths and weaknesses. This however requires detailed analysis the results using much wider test sets. Obtained results however seems to justify the investment needed for further evaluation of the proposed methods and give hope for even better results in the future.

ACKNOWLEDGMENT

We would like to thank Piotr Kopa-Ostrowski for help with the implementation of the algorithms.

REFERENCES

[1] T. Berners-Lee, J. Hendler, O. Lassila *et al.*, "The semantic web," *Scientific american*, vol. 284, no. 5, pp. 28–37, 2001.

[2] Y. Ding, D. Fensel, M. Klein, and B. Omelayenko, "The semantic web: yet another hip?" *Data & Knowledge Engineering*, vol. 41, no. 2-3, pp. 205–227, 2002.

[3] A. Maedche and S. Staab, "Ontology learning for the semantic web," *Intelligent Systems, IEEE*, vol. 16, no. 2, pp. 72–79, 2001.

[4] K. Goczyła, T. Grabowska, W. Waloszek, and M. Zawadzki, "The knowledge cartography–a new approach to reasoning over description logics ontologies," *SOFSEM 2006: Theory and Practice of Computer Science*, pp. 293–302, 2006.

[5] H. Sun, W. Fan, W. Shen, and T. Xiao, "Ontology-based interoperation model of collaborative product development," *Journal of Network and Computer Applications*, vol. 35, no. 1, pp. 132–144, 2011.

[6] D. Vallet, M. Fernández, and P. Castells, "An ontology-based information retrieval model," *The Semantic Web: Research and Applications*, pp. 103–110, 2005.

[7] J. Sowa, *Principles of semantic networks*. Morgan Kaufmann, 1991.

[8] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data – the story so far," *International journal on semantic web and information systems*, vol. 5, no. 3, pp. 1–22, 2009.

[9] A. Gomez-Perez, M. Fernández-López, and O. Corcho, *Ontological engineering*. Springer Heidelberg, 2004, vol. 139.

[10] L. Specia and E. Motta, "Integrating folksonomies with the semantic web," in *The semantic web: research and applications*. Springer, 2007, pp. 624–639.

[11] R. Studer, V. R. Benjamins, and D. Fensel, "Knowledge engineering: principles and methods," *Data & knowledge engineering*, vol. 25, no. 1, pp. 161–197, 1998.

[12] M. Ruiz-Casado, E. Alfonseca, and P. Castells, “Automatic assignment of wikipedia encyclopedic entries to wordnet synsets,” in *International Atlantic Web Intelligence Conference*. Springer, 2005, pp. 380–386.

[13] Ł. Borek and J. Szymański, “Semantyczne znacznikowanie artykułów wikipedii synsetami słownika wordneta,” *Zeszyty Naukowe Wydziału Elektroniki, Telekomunikacji i Informatyki Politechniki Gdańskiej*, vol. 10, 2010.

[14] S. P. Ponzetto and R. Navigli, “Knowledge-rich word sense disambiguation rivaling supervised systems,” in *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, 2010, pp. 1522–1531.

[15] E. Niemann and I. Gurevych, “The people’s web meets linguistic knowledge: automatic sense alignment of wikipedia and wordnet,” in *Proceedings of the Ninth International Conference on Computational Semantics*. Association for Computational Linguistics, 2011, pp. 205–214.

[16] S. Brin and L. Page, “The anatomy of a large-scale hypertextual web search engine,” *Computer networks and ISDN systems*, vol. 30, no. 1-7, pp. 107–117, 1998.

[17] E. Agirre and A. Soroa, “Personalizing pagerank for word sense disambiguation,” in *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2009, pp. 33–41.

[18] R. Korytkowski and J. Szymański, “Collaborative approach to WordNet and Wikipedia integration,” in *The Second International Conference on Advanced Collaborative Networks, Systems and Applications, COLLA*, 2012, pp. 23–28.

[19] J. Szymański, “Words context analysis for improvement of information retrieval,” in *Computational Collective Intelligence. Technologies and Applications*. Springer, 2012, pp. 318–325.

[20] J. Szymański and W. Duch, “Self organizing maps for visualization of categories,” in *Neural Information Processing*. Springer, 2012, pp. 160–167.

[21] The OpenSearch Project, https://github.com/dewitt/opensearch, online, accessed 15.05.2019.

[22] T. Boiński, “Game with a purpose for mappings verification,” in *Computer Science and Information Systems (FedCSIS), 2016 Federated Conference on*. IEEE, 2016, pp. 405–409.

[23] J. Szymański and T. Boiński, “Crowdsourcing based evaluation of automatic references between wordnet and wikipedia,” *International Journal of Software Engineering and Knowledge Engineering*, in press.

[24] L. Von Ahn, “Games with a purpose,” *Computer*, vol. 39, no. 6, pp. 92–94, 2006.

[25] Wikipedia, https://en.wikipedia.org/wiki/Wikipedia:Size_of_Wikipedia, online, accessed 15.05.2019.

[26] MediaWiki Action API, https://www.mediawiki.org/wiki/API:Opensearch, online, accessed 15.05.2019.