Electronic version of an article published as International Journal of Software Engineering and Knowledge Engineering, vol. 29, no. 03, 2019, pp. 317-344. Article DOI: 10.1142/S0218194019500141

International Journal of Software Engineering and Knowledge Engineering

CROWDSOURCING BASED EVALUATION OF AUTOMATIC REFERENCES BETWEEN WORDNET AND WIKIPEDIA

JULIAN SZYMAŃSKI
Faculty of Electronics, Telecommunications and Informatics
Gdańsk University of Technology
11/12 Narutowicza Street, 80-233 Gdańsk, Poland
julian.szymanski@eti.pg.edu.pl
http://<julian.eti.pg.gda.pl>

TOMASZ BOIŃSKI
Faculty of Electronics, Telecommunications and Informatics
Gdańsk University of Technology
11/12 Narutowicza Street, 80-233 Gdańsk, Poland
tobo@eti.pg.edu.pl
http://<tboinski.eti.pg.gda.pl>

Received (Day Month Year)
Revised (Day Month Year)
Accepted (Day Month Year)

The paper presents an approach to build references (also called mappings) between WordNet and Wikipedia. We propose four algorithms used for automatic construction of the references. Then, based on an aggregation algorithm, we produce an initial set of mappings that has been evaluated in a cooperative way. For that purpose we implement a system for the distribution of evaluation tasks, that have been solved by the user community. To make the tasks more attractive, we embed them into a game. Results show the initial mappings have good quality, and they have also been improved by the community. As a result, we deliver a high quality dataset of the mappings between two lexical repositories: WordNet and Wikipedia, that can be used in a wide range of NLP tasks. We also show that the framework for collaborative validation can be used in other tasks that require human judgments.

Keywords: WordNet; Wikipedia; lexical resources integration; natural language processing; Game with a Purpose

1. Introduction

In the World Wide Web information is mostly stored in the form of a text written in natural language. Excluding the multimedia data, WWW can be seen as a dynamic repository of text documents. Searching for relevant information is performed by retrieving web pages that contain words specified by the user. Despite successful applications of keyword-based retrieval, the growing amount of textual information requires more efficient methods to increase precision of the search results [1]. Most

of the textual resources in WWW are unstructured, thus they are difficult to process with computers. Due to that, a lot of effort is put into developing technologies that may help automatically extract and process knowledge from that overwhelming set of information. To make this effort successful it is necessary to provide an elementary machine-readable linguistic knowledge base that allows one to process the text efficiently.

One of the directions to make WWW resources processable by machines is the Semantic Web initiative [2]. The idea aims at extending the Web with meta data to support automatic processing of its content [3] [4]. Semantic here is introduced by annotating words, pages or other Web resources, with references (links) to external structuralised knowledge repositories. An example of such repositories are ontologies [5] that bring a knowledge model typically based on logic. The approach requires that the ontologies contain large amounts of formalized data, and instantly evolve with the culture and language. It can only be achieved with, at least partial, automation of their construction. Although there are promising results, usage of ontologies is mainly domain-oriented, e.g. [6]. Especially for information retrieval in WWW [7] their wide-scale applications haven't been shown yet. This is caused by the fact that the logic is too formal to capture all natural language nuances. Also, the complex knowledge model behind the ontological approach is not as intuitive as hyperlinks. This causes that ontologies are not used by the wide community of web users, and this paradigm did not initiate a snowball effect, yet.

As a result, more elastic and less formal modeling is applied to provide linguistic background knowledge for natural language processing. The most popular are Semantic Networks [8] based on typed relations between concepts representing word meanings. The lack of large-scale and high-quality Semantic Networks is the main obstruction in the development of natural language technologies.

Current trends in developing technologies for publication of resources is Linked Data [9]. In this paradigm a large number of different approaches for linking the data can be selected. They range from the strict logic-based methodologies based on Ontologies [10] to less formal Folksonomies [11]. The common assumption of these approaches is that interlinked data will be more useful. Providing additional references will increase availability of the data for proper machine-based interpretation. Linking the resources, ideally with named relations, provides a model of knowledge that is based on typed references [12] that provides representation space for computing the language.

To make this approach successful, the challenge is to construct a wide range of meaningful links between resources. The main problem here is their construction. Linking the resources by hand is slow, thus this approach usually does not provide a high enough number of relations. This process may be partially completed automatically, but on the other hand the automatically introduced links may not have high quality. This is caused by the limited applicability of natural language processing algorithms, that do not provide a framework for language understanding, but only allows one to statistically interpret its meaning. Due to that, in our research

we combine both approaches: automatic introduction of initial links, and then their manual completion. As manual links creation and verification is a troublesome task, requiring a high amount of time and effort, we resort to a crowdsourcing approach. Most of the work is being done by a large group of volunteers while performing other activities, like playing games, using so called Games with a Purpose [13].

Despite a lot of effort made to provide a valid mapping between multiple resources, especially Wikipedia and WordNet, no definite mapping corpora was proposed. The goal of this paper is to present an approach we used to develop existing lexical resources through their integration; as well as a framework for evaluation and improvement of the results. In particular, the algorithm for automatic integration of WordNet and Wikipedia through creating links (mappings) between synsets and articles is introduced. Since it is not possible to build 100% accurate mappings entirely automatically, a collaborative crowdsourcing-based approach for evaluation and improvement of initial links has been proposed. As a result, we provide a database of links that interconnect WordNet and Wikipedia resources with Precision gaining over 90%.

The article is constructed as follows: Section 2.1 reports the state of the art related to Wikipedia – WordNet integration, Section 2.2 presents the idea of crowdsourcing and Games with a Purpose. Section 3 describes our method of automatic links creation, and describes evaluation methodology. In Section 4 we provide means of collaborative evaluation of the automatic integration, and Section 5 presents the obtained results. The conclusions and proposed future work has been presented in the last section.

2. Related work

2.1. *Wikipedia WordNet Integration*

Wikipedia and WordNet databases are freely available. They are developed independently and present different points of view on human knowledge. At the moment, there isn't any project integrating them completely. That automatic integration would be a step into building a common, easily accessible lexical ontology, which would improve processing of the textual information providing conceptual space for natural language representation. The benefits from such an ontology are inestimable, as it could impact on the organization of WWW resources providing that their content be more explicit and comprehensible by computers. It could streamline the processes of retrieval, management, translation and processing of textual data. Understanding the benefits of WordNet and Wikipedia integration, several projects were developed. Despite showing promising results, none of them were implemented on a large scale, and to the best of our knowledge, did not provide a wide, publicly available database of mappings between synsets and articles.

Ruiz-Casado et al. in their work [14] tag Wikipedia articles with WordNet synsets. They use Simple Wikipedia, which is a version designated for people learning English, containing less articles and using only basic vocabulary. In their ap-

proach, they apply a disambiguation algorithm based on the Vector Space Model to determine similarity between an article and a synset. They ran the algorithm against 1,841 articles, 33% of which were not matched with WordNet synsets, 34% were matched with exactly one synset and 33% required disambiguation. In the case of articles which did not require disambiguation the Precision was 98%, and 84% in the other case.

The reported results were satisfactory. However, we did not expect to come close to that level when applying the algorithm to the full Wikipedia, because of the significant difference in the number of articles and their complexity. Therefore, we decided to take a bit different path. In our initial approach the automatic integration of Wikipedia and WordNet has been based on the word co-occurrence analysis. The approach is described with pseudo code in Algorithm 1. The analysis is performed between a synset definition and the first paragraph of a Wikipedia article [15]. The obtained results (39.51% and 49.28% quality depending on the method) evaluated for 200 test mappings indicate the method can be useful, but it requires the additional contribution of humans.

The YAWN project [16] converts the well-known and widely used Wikipedia collection into an XML corpus enriched with semantically self-explaining tags. The annotation process exploits categorical information in Wikipedia and additional information from lists, tables, links to other Wikipedia and external Web articles. Each generated XML document consists of three parts:

- the preamble that sets the character encoding and includes a pointer to an XSLT for presenting the page.
- the article element with its header child, which in turn has children that specify meta data like the page title and id, the last revision, and the categories of the page.
- the body child of the article element that contains the XML representation of the page's content

It also contains extra semantic annotations, connected with appropriate entries of the WordNet. The separate labels are obtained from:

- categories – the majority of Wikipedia articles are assigned to one or multiple categories. Every conceptual category has to be linked to a corresponding WordNet synset.
- lists – which are an extensive, manually created and therefore high-quality source of information.
- templates invocation – which are a rich source of semantics.

The derived annotations are added to the article right after the article element. Each of the annotations contain a specific tag name derived from the WordNet synset, that corresponds to the annotated concept. The tags are augmented with the confidence factor, the ID of the WordNet concept, and the source of the annotation (category, list, template invocation).

Algorithm 1 Pseudo code for our initial approach

```
for word do
    synsetList ← get all WordNet senses for the given word
    articleList ← get all Wikipedia articles for the given word
    for synset in synsetList do
        POSList=tagPOS(synset.definition) ; perform part of speech tagging
        steam(synset.definition) ; perform steaming
        for tokenId =0; token.Id==POSList.length do
            if POSList(tokenId).type in (noun, verb, adjective, adverb) then
                wordList += steam(tokenId)
            end if
        end for
        synset.tokens=wordList
    end for
    POSList=tagPOS(article.content) ; perform part of speech tagging
    steam(article.content) ; perform steaming
    for tokenId =0; token.Id==POSList.length do
        if POSList(tokenId).type in (noun, verb, adjective, adverb) then
            wordList += steam(tokenId)
        end if
    end for
    article.tokens=wordList
    for synset in synsetList do
        for article in articleList do
            TokensNumber = FindSameTokens(synset.tokens,article.tokens)
            if synset.MaxNumberOfTokens < TokensNumber then
                synset.Mapping == article
                synset.MaxNumberOfTokens = TokensNumber
            end if
        end for
    end for
end for
```

The next project called YAGO is an ontology constructed using Wikipedia and WordNet [17]. Its resources consist of over 2 million objects and 20 million of related facts extracted using text mining algorithms. The project managed to construct around 15,000 direct mappings between WordNet synsets and Wikipedia articles in an automatic way [18] but they are not available separately in the whole YAGO system.

YAGO is a light-weight and extensible ontology with high coverage and quality. It is built on entities and relations and currently contains over 1 million entities and 5 million facts, according to the assurance of authors. This includes the Is-

A hierarchy as well as non-taxonomic relations between entities. The facts have been automatically extracted from Wikipedia and unified with WordNet, using a carefully designed combination of rule-based and heuristic methods. The resulting knowledge base is a major step beyond WordNet: in quality by adding knowledge about individuals like persons, organizations, products, etc. with their semantic relationships, and in quantity by increasing the number of facts by more than an order of magnitude. YAGO is based on a logically consistent model, which is decidable, extensible, and compatible with RDFS.

The organization of YAGO ontology allows you to obtain information through a number of queries, for example: a question about writers living in given years, or connections between two events or persons. Wikipedia has also been used as a source of data for sense annotations. Starting with the hyperlinks available in Wikipedia, a sense annotated corpora has been generated. The corpora can be used for building accurate and robust sense classifiers. Evaluation of constructed annotations has been performed on the Wikipedia-based sense-tagged corpus generated from a subset of Senseval [19] and it shows their high reliability. The quality of a sense-tagging classifier built on this data set exceeds, by a large margin, the accuracy of an informed baseline that selects the most frequent word sense by default. The sense-tagged corpus has been built in three main steps: First, all the paragraphs in Wikipedia that contain an occurrence of the ambiguous word as part of a hyperlink name have been extracted. Also, paragraphs based on the Wikipedia paragraph segmentation, which typically lists one paragraph per line, have been selected. To focus on the problem of word sense disambiguation, rather than named entity recognition [20], named entities have been explicitly avoided by considering only those word occurrences that are spelled with lower-case. Next, all the possible labels for the given ambiguous word, by extracting the leftmost component of the links, has been gathered. For instance, in the pipped link [musical_notation—bar], the label [musical_notation] was extracted. In the case of simple links (e.g. [bar]), the word itself can also take the role of a valid label, if the page it links to was not determined as a disambiguation page. Finally, the labels were manually mapped to their corresponding WordNet sense, and a sense-tagged corpus was created. This mapping process is fast, as a relatively small number of labels is typically identified for a given word. Using a set of sense-annotated examples for a given ambiguous word, a word sense disambiguation system automatically learns a disambiguation model that can predict the correct sense for a new, previously unseen, instance of the word. The disambiguation algorithm starts with a preprocessing step, where the text is tokenized and annotated with part-of-speech tags. Collocations are identified using a sliding window approach, where a collocation is defined as a sequence of words that forms a compound concept defined in WordNet. Next, local and topical features are extracted from the context of the ambiguous word. Specifically, they use the current word and its part-of-speech, a local context of three words to the left and right of the ambiguous word, the parts-of-speech of the surrounding words, the verb and noun before and after the ambiguous words, and a global context

implemented through sense-specific keywords determined as a list of (at most five) words occurring at least three times in the contexts defining a certain word sense. Beside CYC [21] YAGO is one of the richest ontologies up to now. It has been used in many applications, where one of the most spectacular is a competition between humans and machines in the television game named "Jeopardy!" that has been won by IBM Watson [22]. Despite its light-weight data model that employs WordNet and Wikipedia resources, it does not provide an explicit database of links between them.

Paper [23] reports an approach to integrate the Wikipedia category system and WordNet synsets. Proposed methodology takes as input a Wikipedia taxonomy. First, it associates a synset with each of Wikipedia categories in the taxonomy and then disambiguates it using a WordNet graph to identify the most relevant synset for each Wikipedia category. Next, it restructures the taxonomy in order to increase its alignment with the WordNet subsumption hierarchy to identify the most relevant synset for each Wikipedia category. The approach allows you to overcome the sparseness of Wiki Taxonomy and complete category system, with the information provided by WordNet. The results have been reported to be provided on-line[a] but during the writing of this work the URL has been unavailable.

Integration of external ontologies with Wikipedia has been the subject of many studies e.g. [24] report approach for linking ontology classes to Wikipedia articles. In the experiment they report high accuracy achieved in the task of integrating Wikipedia with Music Ontology. Also [25] reports the good results of integrating Wikipedia with textual resources where concepts based on Wikipedia articles have been used for creating text representation suitable for usage in machine learning.

Automatic alignment of WordNet synsets and Wikipedia articles has been used to obtain a sense inventory. In the approach [26] for each WordNet synset a set of related Wikipedia articles has been found. Then, in a second step, a determination of which article is a valid alignment has been performed. The approach report 0.78 F_1 – measure based on a comprehensive reference dataset consisting of 1,815 manually annotated sense alignment candidates. The resource has been made publicly available[b], the latest database deploy is dated March 2012 and the project is still under development, as it reports "Due to a technical problem in the candidate extraction process, not all candidates were considered for each synset, i.e. some correct alignments are missing. The alignments which are present in the file should be correct".

2.2. *Crowdsourcing and Games with a Purpose*

Despite the evolution of computer systems and algorithms, there are problems that cannot (or are hard to) be solved automatically. A task like mappings creation

[a]http://www.eml-research.de/nlp/download/wikitaxonomy.php
[b]http://www.ukp.tu-darmstadt.de/data

requires alternative approaches. When a problem is numerically solvable, but simply requires a lot of computational power, a volunteer computing model [27] can be applied. In this model the users donate the power of their machines to solve the task. Unfortunately, some of the problems cannot be successfully turned into a computer algorithm [13]. In such cases we can use heuristics. In most cases they provide good enough results. In some solutions, e.g. when creating a mapping corpus, we require near perfect correctness of the algorithm output. In such cases a crowdsourcing [28] approach can be used. The task in this case can be both algorithmic and non-algorithmic and the user is encouraged to take part in the process for some type of gratification.

One of the forms of crowdsourcing is so called human-based computation (HBC) [29, 30]. In this approach the users solve a problem based on their knowledge and experience.

The pure crowdsourcing approach does not involve computers in the computation process – the problem is solved by humans. In HBC however part of the problem can be solved by a computer e.g. sub-problem organization, distribution and retrieval of results, some calculations using heuristics. The human task is either final creation, or verification of the results [31].

Crowdsourcing gained some popularity. In most cases the platforms like Amazon Mechanical Turk[c] give the users some small financial gratitude for every task solved. The money is paid by the problem supplier. Other solutions, like Snapshot Serengeti[d] project [32], are based purely on the will of the community to help in solving the problem. Both of these approaches proved to be successful.

The idea of gaining users without actual payment led Luis von Ahn to the introduction of Game With A Purpose (GWAP) [13]. In GWAPs the entertainment coming from playing the game is the reward itself – no other gratification is allowed [33]. The game, however, contains elements helping with solving real problems by the players; usually designed as the games' objectives.

In his work Luis von Ahn distinguished three types of GWAPs:

- output-agreement game – the players are presented with identical data and are expected to provide identical responses. An example of such a game is the ESP Game [34] allowing image tagging by keywords during a time-limited window. The images could be tagged with matching keywords. Later on, based on the original success [13], Google extended the idea in Google Image Labeler[e] which, until being shut down in 2011, helped to improve Google Graphics service.
- inversion-problem game – the players are divided into two groups where one group has to guess the input based on the information from the other group. Examples of this type of GWAPs are Phetch [35] or Peekaboom [36]

[c]https://www.mturk.com/mturk/welcome
[d]https://www.snapshotserengeti.org/
[e]http://en.wikipedia.org/wiki/Google_Image_Labeler

games. In this case the players had to either find an identical image, or guess the contents of the image based on the hints given by other players. Furthermore, the input images could be tagged with the hints-based keywords that led to the correct answer.

- input-agreement game – the players have to decide whether they were presented with the same or different input. An example here can be TagATune [37] game, where players described the tune that was played. Once again the tunes were tagged with keywords based on the players' descriptions.

All three types of GWAPs allow performance of tasks that cannot be grasped by a typical algorithmic approach. All of them also need the same thing – a large user base. Only then, the results can be seen as viable. That implies in turn a need to provide an interesting product that will encourage players to play the game and thus participate in the whole process. Von Ahn proposed a few elements that should be included in such games, like time limits, awards in the form of points and achievements, difficulty levels, leader boards, or randomness of input data [33]. He also introduced parameters that indicate the success or failure of given product: how long on average the players play the game, and average number of problems solved per hour. Later on, Simko also introduced the total number of players as an equally important factor [38].

The typical GWAP is a very targeted game, and the game mechanics are strictly connected to the problem. If the users task is, for example, to create mappings, he or she does just that – answers a series of questions that lead to generation of the mappings. Examples of such games are Phrase Detectives[f] or Wordrobe[g].

In some cases the game (or software in general) is used to verify the results of heuristic algorithms. The most popular example of that type (where the users do not even realize the purpose behind the product) is Google reCAPTCHA[h] [39, 40] and Duolingo[i] [41]. In the first case, the users verify previously scanned words and tags for pictures during anti-bot verification, and in the second they provide translations of texts while learning languages. Other examples are more explicit, but still try to hide the task within the game mechanics; like in Infection and The Knowledge Tower[42]. Some cases, like Puzzle Racer [43], try to provide game play distinct from the task presented to the users.

[f] http://anawiki.essex.ac.uk/phrasedetectives/
[g] http://wordrobe.housing.rug.nl/Wordrobe/public/HomePage.aspx
[h] http://www.google.com/recaptcha/intro/index.html
[i] https://www.duolingo.com/

3. Our approach to Wikipedia and WordNet integration

3.1. *Wikipedia pruning*

In the presence of a significant disproportion between the number of articles in Wikipedia and WordNet synsets, there is a need to pre-process Wikipedia and eliminate articles that are unlikely to be matched with WordNet synsets. The approach we took was to query Wikipedia via the Opensearch API[j] with words taken from WordNet. We set a limit to twenty results per query, and found this way a set of 337,104 potentially relevant articles. To limit the number of remote requests in our approach we implemented a local cache that stores results retrieved for a particular search phrase.

We have prepared a series of statistics for the returned data. Almost half of the queries (43.87%) returned a unique result. The limit we use for the results has been reached only for 0.02% of queries. It indicates that we do not lose too much information.

In addition, 78.6% of articles are unambiguous; which when compared to 51% of noun synsets defining only one phrase, is a rather high number. It is partially due to the fact that we recognize ambiguous phrases only if they occur both in WordNet and Wikipedia.

Table 1. Wikipedia pruning: phrases per articles

Phrases	Articles
1	264,959 (78.60%)
2	45,156 (13.40%)
3	14,324 (4.25%)
4	6,076 (1.80%)
5	2,839 (0.84%)
6	1,529 (0.45%)
7 and more	2,221 (0.66%)
Total	337,104

Based on our analysis of WordNet and Wikipedia structure we have implemented an algorithm, which automatically constructs links between these two lexical resources. It is known that not all WordNet synsets can be linked with Wikipedia articles. Many times general terms are not present in Wikipedia. For instance *friend* (a person you know well and regard with affection and trust) is not found in Wikipedia. The closest match we could find was *friendship*. However, more specific terms like *girlfriend* or *boyfriend* could be easily found. It is partially because WordNet is a dictionary whereas Wikipedia is an encyclopedia. For the mappings to be useful,

[j]http://www.mediawiki.org/wiki/API:Opensearch

Table 2. Confusion matrix used for evaluation of mappings

evaluation \ algorithm	created mapping	not created mapping
mapping was possible	TP	FN
	FGM	
mapping couldn't be created	FP	TN

we are less interested in vague matches, and we are looking for exact matches. We also prefer not to create a link, than to set up a wrong one.

3.2. *Results evaluation*

The set of automatically created links we evaluate using two measures taken from the Information Retrieval domain. Precision and Recall have been derived from the modified confusion matrix shown in Table 2.

In the Table 2 we can select a evaluation column where algorithm, for a particular synset, has automatically created the link, or not. On the other hand, when a human evaluates the mapping, she or he can form the link between WordNet and Wikipedia or not – due to the lack of corresponding elements in both resources. Successive rows of the confusion matrix describe the instances when manual construction of the link was possible and not.

Thus, for the evaluation, we account for the following instances:

(1) the algorithm set up the link that could be created. Here we have two cases:
   (a) TP (true positive) – algorithm built proper link
   (b) FGM (false generated mapping) – algorithm created link and it is wrong.
(2) TN (true negative) – the algorithm didn't create the link while it was not possible
(3) FN (false negative) – the algorithm didn't create the link while it was possible
(4) FP (false positive) – the algorithm created the link while it was not possible

The number of occurrences of particular instances allow us to derive Precision (Formula 1) and Recall (Formula 2) measures that describe the quality of automatically constructed mappings. In the experiments we also provide a Coverage measure (Formula 3) that describes the fraction of mappings that we were able to construct while using each of the proposed approaches (Section 3.3).

$$Precision = \frac{|TP|}{|TP| + |FGM| + |FP|} \tag{1}$$

$$Recall = \frac{|TP|}{|TP| + |FGM| + |FN|} \tag{2}$$

$$Coverage = \frac{|TP| + |FGM| + |FP|}{|synsets|} \tag{3}$$

It should be noticed in our system for evaluation of the mappings (Section 4), the user can provide a fuzzy answer. The mapping can be assessed as perfectly acceptable or wrong. To calculate the Precision and Recall measures, we decide both perfect and acceptable evaluations are treated uniformly as correct.

3.3. *Automatic mappings construction*

The final mappings we built aromatically, has been constructed as an aggregation of results obtained from four independent approaches:

3.3.1. *Unique results*

The *unique results* approach was based on the fact that most of WordNet phrases are used only in one synset. If a phrase is unique (it is only related to one synset), and querying Wikipedia returns only one result, then we form a mapping. This approach is described with pseudo code in Algorithm 2.

The evaluation for 200 random synsets has revealed Precision of 0.97 +- 3.34%. That gives us 32,024 linked synsets out of a total 82,115, that is 38.99 % Coverage. This mapping procedure has been completed in 43.4 minutes.

Algorithm 2 Pseudo code for Unique results approach

```
synsetList ← get all synsets from WordNet
for synset in synsetList do
    wordList ← get all words for synset
end for
for word in wordList do
    if synsetList(word).Size == 1 then
        articleCollection ← using Wikipedia API find Wikipedia articles containing word in a title
        if articleCollection.Size == 1 then
            mappingList(synset) = articleCollection(1)
        end if
    end if
end for
```

3.3.2. *Approach based on synonyms*

In the presence of 21.4% synonyms in the pruned Wikipedia and 49% in WordNet synsets, we assumed that if the same article occurs at least twice in the results from

querying Wikipedia with synonym words from WordNet then we build a mapping. This approach has been described with pseudo code in Algorithm 3. Its running time took 63.2 minutes.

Algorithm 3 Pseudo code for approach based on synonyms

```
synsetList ← get all synsets from WordNet
for synset in synsetList do
    wordList ← get all words for synset
    if wordList.Size > 1 then
        for word in wordList do
            articleCollection ← using Wikipedia API find Wikipedia articles containing word in a title
            for article in articleCollection do
                if articleList(article).Size == 0 then
                    articleList.add(article)
                else
                    mappingList(synset) = article
                end if
            end for
        end for
    end if
end for
```

The *synonyms algorithm* has covered 22% of synsets. For 200 of randomly selected, and manually evaluated, mappings we got Precision 0.88 +- 6.43% . That gives us 18,065 linked synsets.

An example of a synset where the algorithm works well is *Harvard, Harvard University [a university in Massachusetts]*. Querying Wikipedia with the *Harvard* phrase returns 14 results whereas *Harvard University* 13 results. Both queries return the *Harvard University* article in the top position in the result list, thus it is recognized as the correct one.

An example of a wrong mapping generated using that approach is shown for synset **commission, delegacy, delegation, mission, deputation** *[a group of representatives or delegates]*. The algorithm built an invalid mapping to the article **Delegation** *[Delegation (or deputation) is the assignment of authority and responsibility to another person (normally from a manager to a subordinate) to carry out specific activities.]*, which is contained in the results for *delegation* and *deputation*. The correct article **Delegate** *[A delegate is a person who speaks or acts on behalf of an organization (e.g., a government, a charity, an NGO, or a trade union) at a meeting or conference between organizations of the same level]* is to be found in the returned results, but it is further down on the list.

3.3.3. *Exact matches*

A third implemented approach builds a mapping whenever an article title and a synset phrase are the same, but only if the phrase was used in no more than one synset. The detailed description has been provided with pseudo code in Algorithm 4.

Algorithm 4 Pseudo code for Exact matches approach

```
1: synsetList ← get all synsets from WordNet
2: for synset in synsetList do
3:     wordList ← get all words for synset
4:     for word in wordList do
5:         articleCollection ← using Wikipedia API find Wikipedia articles containing word in a title
6:         for article in articleCollection do
7:             if compare(article.title, word) == 0 then
8:                 mappingList(synset) = article
9:             end if
10:        end for
11:    end for
12: end for
```

This approach allows us to increase Coverage of the formed mappings, but it should be noted that it could negatively influence the Precision. We have experimented with changing the similarity measure between the Wikipedia article and the word from WordNet (in pseudo code described with function *compare*). To effectively process the imperfect matching that allows us to capture small differences between the search word and article title, we used our algorithm [44] that allows us to faster process the collections of strings, and gives better results than the typically-used Damerau-Levenshtein [45] distance. The initial review of results has shown that indeed this strongly increases the Coverage, but the loss of Precision is significant: thus, we decide to use only exact matches similarity.

As a result of the proposed approach 59% of synsets have been linked with articles with a Precision of 83% +- 7.35% evaluated manually for 200 random samples . That gives us mappings for 48,447 synsets that have been completed in 52.1 minutes.

The strength of this algorithm lies in the fact that 51% of synsets have exactly one sense and define such unique terms as Lycopodium obscurum, Centaurea, Green Revolution, etc.

Among wrong results the **fishbone** *[a bone of a fish]* synset is to be found, which is mapped to the **Fishbone** article *[Fishbone is a U.S. alternative rock band formed in 1979 in Los Angeles, California, which plays a fusion of ska, punk rock, funk, hard rock and soul.]*. To our surprise, manual search did not let us find any

correctly matching article.

3.3.4. *Most-used*

The last proposed approach was based on an assumption, that if we have a ranked list of articles, the first one is correct. In our implementation we use results from the Wikipedia Opensearch API, and for a given search phrase we use as a link with the synset the first article that has been returned at the top of the result list. If a synset has synonyms, then we select the article that appears most frequently, and at the highest positions, among all returned results.

This approach has been described with pseudo code in Algorithm 5. It was tailored for improving the Coverage, it is used as the last-chance trial of mapping construction.

Algorithm 5 Pseudo code for most-used approach

```
1: synsetList ← get all synsets from WordNet
2: for synset in synsetList do
3:     wordList ← get all words for synset
4:     for word in wordList do
5:         articleCollection ← using Wikipedia API find Wikipedia articles containing word in a title
6:         for article in articleCollection do
7:             ArticleHashMap[article] += articleCollection.size - articleCollection(article).position
8:         end for
9:     end for
10:    article = ArticleHashMap.maxValue
11:    mappingList(synset) = article
12: end for
```

The evaluation of randomly selected 200 test mappings confirms that the approach has high Coverage, but it introduces a high number of wrong mappings. As many as 84% synsets have been mapped with a measured Precision of only 17% +- 7.36%. That gives us 68,976 mapped synsets, but with only 11,726 +- 5,047 correct. Due to usage of an external API the running time of the algorithm took 8h 27 min.

3.4. *Automatically generated dataset of links: Aggregation of the results*

Proposed approaches allow us to build automatically a large set of links between articles and synsets. To achieve the set of initial automatic links the results of each of the separate approaches have been aggregated in such a way that they maximize the quality measure.

The best combination of presented algorithms has been selected using a set of 200 test links that maximize the F-measure [46]. The F-measure is a widely used metric for evaluation of results in Information Retrieval, where there is a weighted harmonic mean of Precision and Recall defined with Formula 4.

$$F = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{4}$$

We performed a series of test combinations of the proposed approaches, their results are presented in Table 3.

Table 3. Results of mapping algorithms

Algorithm	Precision	Recall	F-measure	Coverage
Unique results (UR)	0.97	0.38	0.55	0
Synonyms (S)	0.88	0.22	0.35	0
Exact matches (EM)	0.83	0.59	0.68	0
Most used (MU)	0.17	0.84	0.28	0
UR + S + MU	0.37	0.81	0.51	0
UR + S	0.86	0.43	0.58	0
UR + S + EM	0.73	0.68	0.70	0

The results presented in the first part of Table 3 have been ordered according to the Precision of the results they produce. To resolve conflicts during aggregation we use two rules:

(1) If two or more approaches create different links, we take the mapping that has been set up by the approach having higher Precision.
(2) If the same link has been built by two or more approaches, we use that link to construct the mapping.

The last proposed approach (MU) introduces too much noise into the initial automatic result set. As Precision of these results is low, to construct a general purpose dataset, we decide to resign from using the MU approach. If, for some tasks, there is a need to have data with high Recall, the results obtained with the MU approach can be added to the remaining part of unmapped synsets.

The second part of Table 3 shows the results of different combinations of proposed approaches. The last one produced the best results, thus we use it to generate initial results of automatic aggregation, and they have been shown in Figure 1. 60,623 synsets were mapped as a result of the intersection of the approaches Unique Results, Synonyms and Exact matches which is 74% of all noun synsets with a measured Precision of 0.73 +- 8.7%, which is as many as 44,254 +- 5,247 correctly mapped synsets. The final procedure of aggregation of the results has been completed in 2h 35 min.

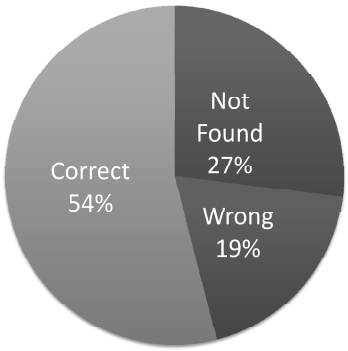

Fig. 1. Initial dataset of automatically created links : results of aggregation of Unique Results, Synonyms and Exact Matches approaches

4. Collaborative evaluation and correction of mappings

Due to the nature of the problem, it is not possible to evaluate and correct formed mappings automatically. To achieve higher precision, and in order to speed up the process of evaluation of existing and creation of missing mappings, we decided to use a crowdsourcing approach.

4.1. *ColabMap*

The first way of grasping what can be achieved using the crowdsourcing approach was the creation of a web page for collaborative work. The ColabMap[k] project enables many users to work simultaneously on linking WordNet and Wikipedia [47, 48, 49, 50] (Figure 2). Their task is to evaluate the correctness of automatically formed mappings, as well as to manually create the new links.

The user needs to login in order to start assessing mappings. The authentication allows the tracking down of already assessed items, so that they are not presented to the same person twice, but to resolve the problem of different opinions from different people. Next, to the user a random synset is displayed. As it is easier to set up a mapping for an unambiguous synset than to set up one containing ambiguous words, for evaluation we select a synset with probability given by a normalized and inverted synset polysemy value. If, for a selected synset, a mapping was set up, a related excerpt from a Wikipedia article is presented. The user needs to choose one of four possible actions: Wrong, Acceptable, Perfect, or Skip. Skip should be chosen if the user does not have enough expertise, or can not find an article (using a keyword-based search engine) that is related to the given synset.

On the other hand, if a mapping does not exist yet, or was evaluated as wrong, the user is asked to build a new mapping. In such a case, a list of recommended

[k]http://kask.eti.pg.gda.pl/colabmap

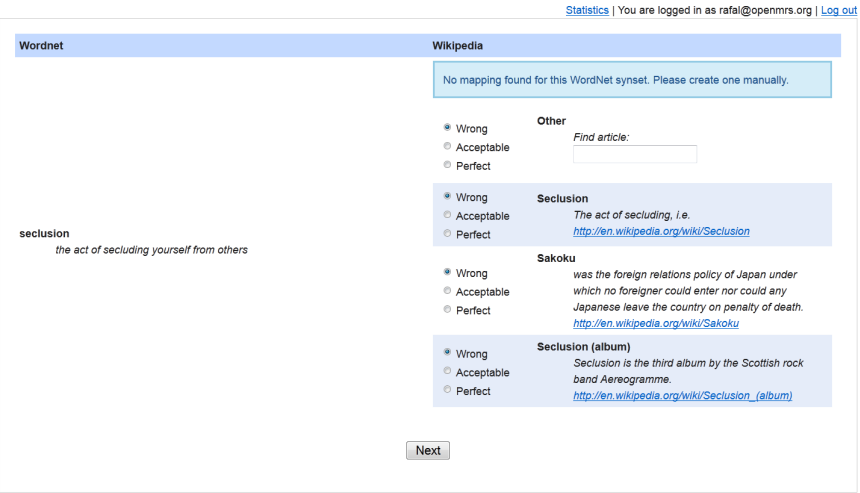

Fig. 2. ColabMap user interface

articles from Wikipedia is presented to the user. There is also an option, which allows searching Wikipedia manually, to find an article that is not found on the list.

The answers of particular users are stored separately, so that if an administrator discovers a malicious user, his or her answers can be easily deleted. The results are presented on the statistics page. One can find there real-time statistics of evaluated and constructed links. There is also a feature which allows the export of mappings in a text format, but it is not yet exposed via the web interface as it is reserved for the system administrator.

The application back-end is written entirely in Java using the Spring framework. All data including WordNet and pruned Wikipedia are stored in the database. For efficiency, all Wikipedia queries and results are cached in the local data base as well. The module for accessing dictionaries and mappings can be easily decoupled from the web application, and used in other applications through a well defined API. It allows the search for terms in both dictionaries making use of the established mappings. The latest database of mappings between WordNet synsets and Wikipedia articles we deployed on the web page of our ComputationalWikipedia project [51] aiming at creating computational representations of Wikipedia [52]. The results of cooperative evaluation allow the increase of final quality of the mapping dataset. Automatically initiated dataset of mappings with most current cooperative corrections has been provided on-line under URL: `http://kask.eti.pg.gda.pl/CompWiki/files/mappings.zip`. The file contains three columns respectively: "WordNet Words", "WordNet Gloss", "Wikipedia URL". The most current dataset contains 82,115 synsets. 50,130 of them have a Wikipedia corresponding link which gains Coverage 61.05%. The 743 mappings evaluated during

testing of proposed approaches (section 3.3.1 – 3.3.4) and 1,778 of the synsets evaluated by system users gives Precision 0.92.

To evaluate synsets without mappings we randomly select 200 synsets from FN and TN sets (described in Table2). In the task of setting up mappings for them, 17 new links have been found. This confirms the Recall of the proposed automatic approaches is high and any additional links can be introduced automatically. The review of these new mappings shows that to form proper mapping, additional knowledge is required. As for automatic setting-up of mappings, we use only the information coded in strings that forms the Wikipedia article titles and synset's synonyms, we can not expect the machine will produce proper results. For most cases where additional linguistic knowledge is required, only humans who understand the language will be able to make such a mapping. Possible extensions of the approaches to automatic mapping through adding external lexical knowledge certainly could increase the results' quality. We argue that the effort for the introduction of such modifications in comparison to the gain will be so small that is better to complete the missing mappings by hand.

4.2. *TGame*

Our second approach was a 2D platform game called TGame[1] [53] ("Tagger Game"). It followed the output-agreement model. The game was designed for an Android platform and was written purely in Java language. To introduce higher replayability, and encourage players, the game contains a point and trophies system based on player in game performance; both in the form of collectibles gathering, like coins or hearts, and in the number of tasks solved. A screenshot from the game can be seen in Figure 3.

Distinctly from other available solutions, we aimed at designing a more general approach. The aim was not to tie the tasks that the player was supposed to do into the game play itself, but on the other hand it couldn't be intrusive. In our approach we decided on following the micro-transaction route. Usually platform games contain some sort of checkpoint system (a restarting place where the player character is moved when killed). Activating the checkpoint requires not only reaching it, but also answering the question tied with the checkpoint. When the answer is correct the checkpoint is activated, and upon subsequent deaths the player restarts at this point instead of the beginning of the current level. A wrong answer results in the checkpoint not being activated. A player has only one approach to checkpoint activation.

Every answer the players give, alongside time taken to read and answer the question, is stored in the database for statistical analysis. The number of correct and incorrect answers gives the information about the correctness of the given

[1]https://play.google.com/store/apps/details?id=pl.gda.eti.kask.tgame,
http://kask.eti.pg.gda.pl/tgame/

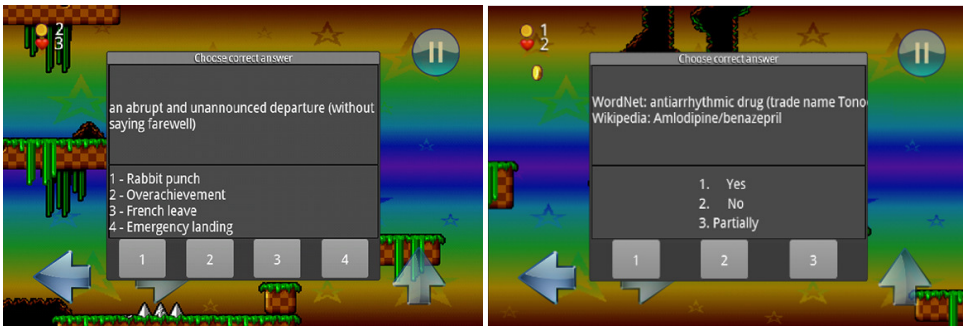

Fig. 3. TGame user interface (left: extended mappings, right: yes/no question)

mapping – if a majority of the players answer in accordance to the database, then we can consider the given mapping as correct. Otherwise, it needs looking into and manual correction.

We also included a possibility of explicit answer reporting. If the player answers wrongly but does not agree with the result, he or she can report the question. This serves as a stronger indication that the mapping in the database is not correct according to the player. This process requires active action from the user, and can be done only once and only for the last answer given. The added burden of question reporting aims at limiting the number of false reports.

4.2.1. *How to ask questions?*

The way of question asking plays an important role in this approach. They have to fit somehow into the game. As stated in previous sections we decided to implement the TGame as a 2D platformer with tasks available during the checkpoint activation. Creating a restart point in the game can be considered here as some form of payment for solving the task. In all cases the difference was in the form of the question, and type and number of available answers.

Extended mappings During the first approach the original Wikipedia-WordNet mappings were extended with three additional "next best" pages from the Wikipedia [53]. The user should then choose an answer from the presented four options where the question was taken from the WordNet (definition of a synset) and the possible answers were the Wikipedia article titles. At first, the 3 other pages were randomly selected from the set of mapped Wikipedia's articles. This was quickly changed, as the randomly selected pages were not related at all to the question [53]. We choose to look up candidate pages using Wikipedia search functionality, e.g. for WordNet synset *ice age, glacial epoch, glacial period* the original mapping was Wikipedia page *Ice age*. After using the Wikipedia search functionality we added 3 more mappings as user choice: *Pleistocene*, *Wisconsin glaciation*

and *Gravettian*. For the tests we randomly selected 233 synsets from our database to limit the time needed to gather the results and verify the viability of the game.

The tests were performed in friends and family model, the software was distributed mainly among students and fellow researchers aged from 20 to 30 years old, in most cases students of computer science at Gdańsk University of Technology. Some participants were also random players that installed the game using Google Play. During the first two months of tests the players gave 3308 answers resulting in 14 answers per question on the average, and around 55 answers daily. The players gave also 625 reports.

This approach showed some limitations, as in some cases the additional pages gathered using the Wikipedia search functionality were very similar in name with the mapped ones, introducing a lot of ambiguity among the potential answers. Selection of random mappings also made the questions vary in terms of difficulty, some requiring expert-level knowledge, e.g. for WordNet synset *Asiatic nut trees: wing nuts* we had 4 options: *Pterocarya*, *Pterocarya fraxinifolia*, *Pterocarya stenoptera* and *Cyclocarya*. This in turn did not match the genre and overall simplicity of the game.

Yes/No questions To mitigate the problems that occurred when using extended mappings we modified the questions asked to Yes/No questions. In this approach the question consists of a pair WordNet definition – Wikipedia article title. The user is than asked a simple question – "yes" or "no". We added the option "not sure" and allowed one to skip the questions to negate the problem of expert level questions. Still, only yes or no answers could be valid; however, if one of the remaining answers is given it is still noted in the database for analysis.

Just as in the previous case we randomly selected a subset of connections from the database for which we generated questions. In this case we created 99 questions for 87 WordNet synsets. Once again the tests were performed in friends and family model. During the first 30 days of tests the players gave 423 answers resulting in 4.27 answers per question on the average, and around 14 answers daily. The players gave also 10 reports.

5. Final Results

The results obtained during the evaluation of the proposed solution are twofold. We obtained a set of verified mapping between Wikipedia and WordNet. Such cooperatively evaluated mappings currently are not publicly available so we believe that our work is a significant contribution to extension of structuralised natural language resources. The mappings might be used in many areas like improvement of human – computer interaction or information retrieval, where it extends the text representation. Furthermore some observations were made on the users' behavior which will prove useful in future implementations.

5.1. *Should we award players?*

ColabMap did not provide any prize for the work, TGame did – in all cases the player was awarded with checkpoint activation upon a correct answer or a report. In the case of Yes/No questions we also introduced trophies and a ranking system where players got points for solving tasks and overall progress in the game. This allowed keeping players with an already familiar product, that otherwise could get bored due to a missing novelty factor.

We performed some additional tests without a prize factor. For that purpose a simple web interface with questions used in TGame was created. The players task was to select one of the extended mappings just as in the game. For the tests a subset of 79 questions was used. The task of the players was simply to ask the questions. In this case we obtained only 250 answers and 50 reports. That gives 3 answers per question and 4 answers daily. The results from the ColabMap manual matching interface (461 questions, 649 answers, 1.4 answer per question and 10.8 answers daily) also shows that better results are obtained when some kind of prize is involved, as in this case players also were not awarded.

5.2. *Dealing with malicious users*

Malicious users are a problem in many fields of social computing like volunteer computing or crowdsourcing. To eliminate such behavior we took the following measures:

- Both TGame and ColabMap, similarly to volunteer computing solutions, rely on multiple answers. Any decisions regarding the mappings are made based on answers from more than one source; and only considers answers that were given at least 5 seconds after the question was shown (as explained in the next Section). This way the malicious user will be outvoted by the other players and automated sending of deliberately wrong answers becomes time-consuming.
- In ColabMap the user has to create an account to be able to do anything. All actions taken by the user are associated with that account. If the user is found malicious it is easy to remove all harm done by that user, as the system allows us to isolate, and potentially remove, all mappings made or verified by the given user.
- TGame does not require users to create an account, however each device is distinguished by a unique identifier allowing you to eliminate answers sent from that device. If the user creates an account than we are able to pinpoint all answers given by the user on all of his or her devices.
- TGame, as the next iteration in verification solution, introduces rewards for doing the verification correctly (check point activation, trophies, user levels etc.). The aim is to encourage users to do the task properly. For malicious users it will be thus harder to progress within the games.
- TGame introduces a reporting process where reports are treated as a strong

Table 4. Number of answers for different type of questions

Question type	Questions	Answers	Answers per question	Reports	Blind shots	Blind shots %
Extended mappings (no prize)	79	250	3.16	50	34	13.6
Extended mappings (with prize)	239	3,308	13.84	625	16	0.48
Yes/No (with prize)	99	423	4.27	10	12	2.84
ColabMap (with prize)	461	649	1.4	–	–	–
Total	878	4,630	5.27	685	62	1.34

indication that the mapping stored in the database is erroneous. Such reports give some reward to the users. In theory they can be used to add higher impact to wrong answers; however, the process itself is not straightforward, and requires some additional actions preventing it from being exploited easily.

The aforementioned measures limit the number of deliberately wrong answers, however automation of malicious user detection is limited to detecting quick answers. In our further works we plan on extending the solution with a reputation system. Known user's answers can be than weighted according to their status. This approach will limit the number of decisions the administrator has to take personally.

5.3. *Answers evaluation*

At the time of writing the TGame was downloaded between 100 and 500 times. Unfortunately Google Play does not provide more exact statistics. All the players, during the test period, gave 4,630 answers in total to 878 questions and defined 649 mappings for 461 WordNet synsets using ColabMap. The total number of answers is shown in Table 4.

As the answers are given by potentially anonymous people with different backgrounds and knowledge, including familiarity with English language, they have to be carefully analyzed. As mentioned before, randomly selected sets of questions on the average required 5 seconds to be read and understood (with answers). Based on that, we decided to discard all "quick" answers being considered as blind shots. That, however, did not have impact on the results as the number of such answers

was very low. In the worst case, the number of discarded answers was no greater than 13.6% of all given answers. However this happened only in the case of extended mappings without prize. When some kind of reward was taken into consideration the number of blind shots was no greater that 3% giving the average percentage of blind shots across all tests no greater that 1.56%. This does not include results given by ColabMap, as the time needed to solve the task was not gathered.

The percentage of blind shots also shows, that rewarding players is a simple yet effective way to encourage people to perform the tasks better. Also, the reward increases the number of answers given. For the Yes/No questions, the lower number of answers can result from missing the novelty factor in the game used – we updated the client with the new forms of questions without introducing new content. This shows that the GWAP approach must be accompanied by a constant stream of new content and replayability. In the long run, the relatively vast amount of content, combined with the ranking system should provide such encouragement.

5.4. *Mappings Update*

Based on the answers that we received from the players we updated the mappings created automatically using ColabMap heuristics. For selecting the best strategy we reevaluated three approaches on when to treat a given mapping as correct for the current data set:

- 75% of the player answers agreed – only 50% of original mappings managed to get enough answers, none of the incorrect mappings were marked as correct,
- at least 50% of player answers agreed – 64% of verified mappings were marked as correct which covered 75% of mappings selected for the test that were originally marked as correct, some false positives were also created,
- the option with most of the answers marked as correct – 74% of verified mappings were marked as correct, and covered 80% of selected mappings selected for the test that were originally marked as correct, some false positives were generated.

The results were identical as in our previous research [53], so in our current solution we decided to once again implement the third approach as it provided the best results. It is worth noting that neither approach allows us to automate the process of database correction. The results obtained allow, however, to pinpoint problematic mappings for verification. Those are mappings that the majority of the answers are different than those obtained using heuristics or the players posted reports for.

The mappings update process is an iterative process. The ColabMap system always presents mappings for the currently selected Wikipedia - WordNet pair. As such both automatically and manually created mappings are provided for verification. In TGame, the current state of the mapping, either automatically generated or modified by the admin, is distributed to the users. In this case, already verified

mappings are re-verified until the administrator explicitly marks it as excluded from verification. Using TGame, as shown in Section 5.3, we also gather multiple answers for a given mapping and the final decision is based on more than one user response.

Using the data gathered, we managed to fix some of the mappings. After the corrections, such mappings did not receive further reports and wrong answers. The current set of mappings after verification can be found online under URL: `http://kask.eti.pg.gda.pl/CompWiki/files/veriffied\_mappings.zip`.

6. Conclusions and future work

Links between WordNet synsets and Wikipedia articles make it possible to use these two resources together for natural language processing. We believe the mappings should improve existing text representations used for processing of a language with machines. The basic assumption in that the task is to provide extended information about words in the written text; and to use it to capture the elementary meaning of the utterances.

The proposed solutions allow creation of such mappings and, which is the most important, ensure their quality. For purely automatically created mappings, even for those with high assumed quality, it is not possible to state the correctness of a given random mapping without actual verification. Manual verification of the whole data set is unfortunately a complex task. With such a framework operational, the mappings authors can trust the initial mappings and add corrections on the fly. Based on the results obtained, the mappings with a majority of answers in accordance to the one created by heuristics can be treated as correct. Only the one pinpointed by the players has to be verified and updated. As such, the data set can be over time updated to reach 100% correctness.

The integration of the resources also opens possibilities to improve WordNet development. We plan to mine Wikipedia [54] structure and automatically introduce new, significant relations to WordNet. It should considerably extend the cross part of speech relations that are especially slimly defined in WordNet. The research made on a simple English Wikipedia presented in [55] shows that it can be a promising direction.

We also plan to extend WordNet sparse synset definitions with extensive articles content. Note that the definitions can be translated into other languages thanks to Wikipedia language links, which also enables multilingual linguistic dictionaries development.

The TGame system is being extended and transformed into a fully fledged crowdsourcing platform for heuristic algorithms result verification. With new clients developed, and generalization of the task formats, the platform could be used for verification of not only mappings but any type of results.

The framework created needs algorithms for at least a semi-automatic mappings update based on user answers. Such an algorithm should take into account additional user attributes like his or her reputation, or the average time taken to answer

the question. This step is crucial in keeping the mappings set in sync with users' answers.

Based on the results obtained in the current research we also plan on grouping the mappings regarding the expected difficulty of words used in mappings to better match the questions to the games implemented. It will also allow us the creation of a quiz-like game with proper difficulty levels. It should not be too difficult but still requiring some knowledge at a higher level, thus prompting players to play again and again.

We also plan on implementing a universal captcha-like component which can eliminate the need of actually convincing users to play the game. Such components are currently widely used in the World Wide Web and users are required to solve them. Such a component requires, however, further verification of currently obtained mappings, as the error ratio in the database has to be low enough for the users to actually pass the verification process.

As an alternative to manually creating relations between concepts, we are also working on methods for the acquisition of linguistic knowledge using an approach based on mining human-machine interactions. We implemented a word game where she or he is thinking about a concept, and the machine tries to guess it. In the game we used a knowledge representation model based on a semantic network. It allows us to calculate the optimal question that most effectively narrows the set of potential results. A machine interaction with a wide range of volunteers playing the game allows us to improve the semantic network structure. As we show in the experiments [56], the consensus reached on a level of commonsense relations between natural language concepts tends to approximate the elementary structure of natural language.

References

[1] J. Szymański, H. Krawczyk, and M. Deptula, Retrieval with semantic sieve, in *Intelligent Information and Database Systems*, ser. Lecture Notes in Computer Science, A. Selamat, N. Nguyen, and H. Haron, Eds. Springer Berlin Heidelberg, 2013, vol. 7802, pp. 236–245.

[2] T. Berners-Lee, J. Hendler, O. Lassila *et al.*, The Semantic Web, *Scientific American*, vol. 284, no. 5, pp. 28–37, 2001.

[3] 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.

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

[5] 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.

[6] 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.

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

[8] J. Sowa, *Principles of semantic networks.* Morgan Kaufmann, 1991.
[9] 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.
[10] A. Gomez-Perez, M. Fernández-López, and O. Corcho, *Ontological engineering.* Springer Heidelberg, 2004, vol. 139.
[11] L. Specia and E. Motta, Integrating folksonomies with the semantic web, in *The semantic web: research and applications.* Springer, 2007, pp. 624–639.
[12] R. Studer, V. R. Benjamins, and D. Fensel, Knowledge engineering: principles and methods, *Data & knowledge engineering*, vol. 25, no. 1, pp. 161–197, 1998.
[13] L. Von Ahn, Games with a purpose, *Computer*, vol. 39, no. 6, pp. 92–94, 2006.
[14] M. Ruiz-Casado, E. Alfonseca, and P. Castells, Automatic assignment of Wikipedia encyclopedic entries to WordNet synsets, *Advances in Web Intelligence*, pp. 380–386, 2005.
[15] J. Szymański and D. Kilanowski, Wikipedia and WordNet integration based on words co-occurrences, *Proceedings of 30th International Conference Information Systems, Architecture and Technology*, vol. 1, pp. 93–103, 2009.
[16] R. Schenkel, F. M. Suchanek, and G. Kasneci, Yawn: A semantically annotated wikipedia xml corpus, *Proceedings of 12th Symposium on Database Systems for Business*, 2007.
[17] F. M. Suchanek, G. Kasneci, and G. Weikum, Yago: A large ontology from wikipedia and wordnet, *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 6, no. 3, pp. 203–217, 2008.
[18] F. Suchanek, G. Kasneci, and G. Weikum, Yago: a core of semantic knowledge, in *Proceedings of the 16th international conference on World Wide Web.* ACM, 2007, pp. 697–706.
[19] R. Mihalcea, T. Chklovski, and A. Kilgarriff, The Senseval-3 English lexical sample task, in *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text.* Barcelona, Spain;, 2004, pp. 25–28.
[20] D. Nadeau and S. Sekine, A survey of named entity recognition and classification, *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.
[21] D. Lenat, Cyc: A large-scale investment in knowledge infrastructure, *Communications of the ACM*, vol. 38, no. 11, pp. 33–38, 1995.
[22] D. A. Ferrucci, Introduction to this is watson, *IBM Journal of Research and Development*, vol. 56, no. 3.4, pp. 1–1, 2012.
[23] S. P. Ponzetto and R. Navigli, Large-scale taxonomy mapping for restructuring and integrating wikipedia. in *IJCAI*, vol. 9, 2009, pp. 2083–2088.
[24] N. Reiter, M. Hartung, and A. Frank, A resource-poor approach for linking ontology classes to wikipedia articles, in *Proceedings of the 2008 Conference on Semantics in Text Processing.* Association for Computational Linguistics, 2008, pp. 381–387.
[25] D. Milne and I. H. Witten, Learning to link with wikipedia, in *Proceedings of the 17th ACM conference on Information and knowledge management.* ACM, 2008, pp. 509–518.
[26] 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.
[27] D. P. Anderson and G. Fedak, The computational and storage potential of volunteer computing, in *Cluster Computing and the Grid, 2006. CCGRID 06. Sixth IEEE International Symposium on*, vol. 1. IEEE, 2006, pp. 73–80.
[28] J. Howe. (2006) Crowdsourcing: A definition. `http://www.crowdsourcing.com/cs/`

2006/06/crowdsourcing_a.html.[Online, accessed: 10.10.2017].

[29] A. Kosorukoff, Human based genetic algorithm, in *Systems, Man, and Cybernetics, 2001 IEEE International Conference on*, vol. 5. IEEE, 2001, pp. 3464–3469.

[30] D. Wightman, Crowdsourcing human-based computation, in *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*. ACM, 2010, pp. 551–560.

[31] J. Simko and M. Bieliková, Games with a purpose: User generated valid metadata for personal archives, in *Semantic Media Adaptation and Personalization (SMAP), 2011 Sixth International Workshop on*. IEEE, 2011, pp. 45–50.

[32] A. Swanson, M. Kosmala, C. Lintott, R. Simpson, A. Smith, and C. Packer, Snapshot Serengeti, high-frequency annotated camera trap images of 40 mammalian species in an African savanna, *Scientific data*, vol. 2, p. 150026, 2015.

[33] L. Von Ahn and L. Dabbish, Designing games with a purpose, *Communications of the ACM*, vol. 51, no. 8, pp. 58–67, 2008.

[34] L. Von Ahn and L. Dabbish, Labeling images with a computer game, in *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 2004, pp. 319–326.

[35] L. Von Ahn, S. Ginosar, M. Kedia, and M. Blum, Improving image search with phetch, in *Acoustics, speech and signal processing, 2007. icassp 2007. ieee international conference on*, vol. 4. IEEE, 2007, pp. IV–1209.

[36] L. Von Ahn, R. Liu, and M. Blum, Peekaboom: a game for locating objects in images, in *Proceedings of the SIGCHI conference on Human Factors in computing systems*. ACM, 2006, pp. 55–64.

[37] E. L. Law, L. Von Ahn, R. B. Dannenberg, and M. Crawford, TagATune: A game for music and sound annotation, in *ISMIR*, vol. 3, 2007, p. 2.

[38] J. Simko, Semantics discovery via human computation games, *Semantic Web: Ontology and Knowledge Base Enabled Tools, Services, and Applications*, p. 286, 2013.

[39] L. Von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum, recaptcha: Human-based character recognition via web security measures, *Science*, vol. 321, no. 5895, pp. 1465–1468, 2008.

[40] M. Foley, "Prove you're human": Fetishizing material embodiment and immaterial labor in information networks, *Critical Studies in Media Communication*, vol. 31, no. 5, pp. 365–379, 2014.

[41] AJT, Lessons from Duolingo's Effort to Support Free Language Learning from Crowdsourcing, `https://digit.hbs.org/submission/lessons-from\ \-duolingos-effort-to-support-free-language-learning-from-crowdsourcing`, 2015. [Online, accessed: 12.05.2017]

[42] D. Vannella, D. Jurgens, D. Scarfini, D. Toscani, and R. Navigli, Validating and extending semantic knowledge bases using video games with a purpose. in *ACL (1)*, 2014, pp. 1294–1304.

[43] D. Jurgens and R. Navigli, It's all fun and games until someone annotates: Video games with a purpose for linguistic annotation, *Transactions of the Association of Computational Linguistics*, vol. 2, no. 1, pp. 449–464, 2014.

[44] J. Szymański and T. Boiński, Improvement of imperfect string matching based on asymmetric n-grams, in *Computational Collective Intelligence. Technologies and Applications*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, vol. 8083, pp. 306–315.

[45] F. J. Damerau, A technique for computer detection and correction of spelling errors, *Commun. ACM*, vol. 7, pp. 171–176, March 1964.

[46] G. Hripcsak and A. Rothschild, Agreement, the f-measure, and reliability in infor-

mation retrieval, *Journal of the American Medical Informatics Association*, vol. 12, no. 3, pp. 296–298, 2005.

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

[48] J. Szymański, Mining relations between Wikipedia categories, in *Networked Digital Technologies*. Springer, 2010, pp. 248–255.

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

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

[51] J. Szymański et al. (2012, Jun.) Computational Wikipedia project. `http://kask.eti.pg.gda.pl/CompWiki/index.php?page=wordnet`.

[52] J. Szymański and W. Duch, Representation of hypertext documents based on terms, links and text compressibility, *Proceedings of ICONIP*, pp. 282–289, 2010.

[53] 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.

[54] O. Medelyan, D. Milne, C. Legg, and I. Witten, Mining meaning from wikipedia, *International Journal of Human-Computer Studies*, vol. 67, no. 9, pp. 716–754, 2009.

[55] M. Ruiz-Casado, E. Alfonseca, and P. Castells, Automatising the learning of lexical patterns: An application to the enrichment of wordnet by extracting semantic relationships from wikipedia, *Data & Knowledge Engineering*, vol. 61, no. 3, pp. 484–499, 2007.

[56] J. Szymański and W. Duch, Context search algorithm for lexical knowledge acquisition, *Control and Cybernetics*, vol. 41, no. 1, pp. 81–97, 2012.