Expert Systems With Applications 61 (2016) 327–342

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Identification of category associations using a multilabel classifier

Julian Szymański*, Jacek Rzeniewicz

Faculty of Electronics, Telecommunications and Informatics, Department of Computer Architecture, Gdańsk University of Technology, Poland

ARTICLE INFO

Article history:
Received 11 January 2016
Revised 24 May 2016
Accepted 27 May 2016
Available online 1 June 2016

Keywords:
Wikipedia
Associations mining
Semantic networks
Categorisation
SVM

ABSTRACT

Description of the data using categories allows one to describe it on a higher abstraction level. In this way, we can operate on aggregated groups of the information, allowing one to see relationships that do not appear explicit when we analyze the individual objects separately. In this paper we present automatic identification of the associations between categories used for organization of the textual data. As experimental data we used a network of English Wikipedia articles and their associated categories, that have been preprocessed by a dedicated filtering method for noise reduction. The main contribution of the paper is the introduction of the method based on supervised machine learning for mining relations between these categories. We describe existing in the literature category proximity metrics as well as introduce three new ones, based on observing the properties of a multilabel Support Vector Machine classifier. The first metric uses classifier predictions, the second uses its errors, and the third is based on its model. Comparison to the existing state-of-the-art methods, and to manual assessments, confirm that the proposed methods are useful and are more flexible than typical approaches. We show how different metrics allow us to introduce new significant relations between categories. Aggregated results of mining categories' associations have been used to build a semantic network that shows a practical application of the research. The proposed method for finding associations can be extended with using other approaches than SVM classification, and can find (other than presented in the paper) applications for mining categories in text repositories. Eg.: it can be used for extending the prediction of the rating in recommender systems or as a method of missing data imputation.

1. Introduction

Wikipedia is currently the largest encyclopedia in the world. Its English version alone contains over fifty times more entries than Encyclopedia Britannica, the second largest English language encyclopedia. As of March 2014, English Wikipedia contains over 4.6 million pages[1] and continues to expand, without losing its impact.

Apart from being a rich source of information readable for humans, Wikipedia is a tremendous data repository that is the interest of many researches related to knowledge engineering. One can download it as a whole – complete dumps[2] are published regularly – and subject it to any experiments imaginable. In the last years a number of interesting research projects have been conducted with Wikipedia, for example the SGI Project[3] where links, events and dates were analyzed to visualize the world's history in the last two centuries, or the one using Wikipedia to compute semantic relatedness between words or texts (Gabrilovich & Markovitch, 2007).

Automatic processing of Wikipedia articles often requires a category similarity metric (Chen, Ma, & Zhang, 2009). For example, this was the case in the approach to topic classification described in (Farina, 2010), as well as in the semantic relatedness algorithm WikiRelate! (Strube & Ponzetto, 2006). In both approaches, similarity between categories was approximated by the distance in a category graph. However, this approach does not yield very good results. First of all, it lacks stratification – the number of category pairs assigned the highest similarity (lowest distance) equals the number of edges in the graph. Secondly, such a metric yields a fair fraction of false positives – vaguely connected categories are marked as similar. This is caused by the fact that links between categories, already marked in Wikipedia are not strictly taxonomic, and often short paths are observed for utterly unrelated classes.

Other applications that would be possible with a good similarity metric pertain to Wikipedia analysis and visualization. For example, clustering performed with a similarity measure that goes beyond a category graph could certainly bring some interesting

* Corresponding author.
E-mail addresses: julian.szymanski@eti.pg.gda.pl (J. Szymański), jacek.rzeniewicz@eti.pg.gda.pl (J. Rzeniewicz).

[1] http://stats.wikimedia.org/EN/TablesWikipediaEN.htm
[2] http://dumps.wikimedia.org/enwiki/
[3] http://www.sgi.com/go/wikipedia/

http://dx.doi.org/10.1016/j.eswa.2016.05.039
0957-4174/

results. Visualisation of Wikipedia edits, performed with a state-of-the-art similarity metric algorithm (Holloway, Bozicevic, & Börner, 2007), confirms this point.

More aspects of similarity metric application are related to Wikipedia's users and editors. As for users – browsing the encyclopedia could become more engaging when the relevant category links to content that is thematically related. In the case of editors, the way they manage the category system could be improved. Currently it is the editors' job to create category nodes, and links between them. Because of its large scale, the category graph is incomprehensible for one person, and it takes time until a new category is linked with similar ones. With a good similarity metric, an editor could receive recommendations about classes that might be related with the currently edited page or category.

The main goal of the research presented in this paper is to develop approaches to computing the similarity between categories that organize textual repository. A total of five metrics were implemented, evaluated and compared. Two of them were based on ideas published in other works, described in Section 2. The first one is based on assignments of articles to categories; in the second one, similarity is computed for classes represented by words contained in related articles. These metrics form a baseline for comparison of metrics introduced by us that have been based on observing the properties of a multilabel support vector machine (SVM) (Hearst, Dumais, Osman, Platt, & Scholkopf, 1998) classifier: first based on its predictions, second on errors, and third on its model. Section 5 describes results of the experiments conducted in order to evaluate and compare these metrics. Evaluation was performed in two ways. First, original hierarchic relationships defined in Wikipedia were used as a reference set. For each of the methods it was measured how well it restores those relations. Second, manual assessment of some of the results yielded by the metrics was performed.

The secondary contribution of research presented in this paper is the demonstration of an approach showing how Wikipedia can be automatically transformed into a semantic network. For this, relationships between categories obtained using four selected approaches to compute categories' similarity were combined. Concepts and relations of the network were exported and used in the visualisation presented in the last section.

2. Related work

2.1. Association rule mining

The problem of finding relations between elements of large datasets has been studied since the early nineties. One of the major catalysts of this area of data mining was barcode technology popularization, which allowed big retail stores to automatically collect past transactions data (basket details). (Agrawal, Imieliński, & Swami, 1993) designed an efficient algorithm for mining association rules between sets of items. For example, given sales data obtained from a large retailing company, the following rule was found: `Children's Hardlines` ↦ `Infants and Children's wear` which might be an indicator for shop managers that products belonging to these two categories should be placed close to each other. In the above case, the rule contained only one item in both antecedent and consequent. However, the algorithm also guarantees to find rules containing larger itemsets, e.g.: `Cheese, Ground beef, Ketchup` ↦ `Burger buns`.

Even though the worst case complexity is 2^m, where m denotes number of all items, in practice the algorithm runs much faster because basket sizes are most usually limited by some small number. It is also worth noting that the association rule mining algorithm produces a directed graph, so $A \to B$ does not necessarily mean $B \to A$. A number of methods have been proposed since (Agrawal et al., 1993) that improved its efficiency (Agrawal & Srikant, 1994; Brin, Motwani, Ullman, & Tsur, 1997).

With the transition of sales to the Internet, baskets are often replaced by session logs: one itemset would contain products related to pages a user visited within a single session. What is more, online stores allow for returning customer identification (e.g. using cookies). Thus not only more data is available to process, but also a new field of recommender systems emerged. Association rules mining was adapted in the 1:1Pro system to construct personal profiles based on past activity of individual customers (Adomavicius & Tuzhilin, 2001). Also a method described in (Lin, Ruiz, & Alvarez, 2000) makes use of association rules in order to come up with personalized user recommendation.

2.2. Mining direct relationships between Wikipedia categories

(Holloway et al., 2007) proposed an approach to computing relationships between Wikipedia categories grounded on an idea similar to association rules mining. In this solution articles correspond to baskets, and categories assigned to an article form a single itemset. However there are two simplifications in the proposed approach. First, relationships are calculated only between individual categories. Second, the resulting graph is undirected. Similarity between categories C_i, C_j is specified using cosine Formula 1.

$$\cos_{i,j} = \cos_{j,i} = \frac{\sum_{k=1}^{n} A_k C_i \cdot A_k C_j}{\sqrt{\sum_{k=1}^{n} A_k C_i \cdot \sum_{k=1}^{n} A_k C_j}} \tag{1}$$

where $A_k C_i$ equals 1 when article A_k is assigned to C_i and 0 otherwise. The Formula 1 can be geometrically interpreted as a cosine between vectors in an n-dimensional space, with each dimension representing one Wikipedia article.

An interesting property of Wikipedia categories is due to the imposed naming conventions[4]. Category names must follow a strict structure defined for various types of categories, like topic categories (*Law, Civilization*) or set categories (*Writers, Villages in Poland*). (Nastase & Strube, 2013) inferred new relationships between categories from their names. For example, category named *Mixed martial arts television programs* is a direct concatenation of other two categories *Mixed martial arts* and *Television programs*, therefore there is a premise that perhaps these categories are related.

A somewhat different approach to computing a similarity metric for Wikipedia categories is to infer it from similarities between articles. This solution can yield various results depending on the assumed articles similarity function. A widely used technique allowing one to compute similarity between documents, is by representing them in a Vector Space Model (Salton, Wong, & Yang, 1975) using words as features. A metric based on such representation of articles was implemented in (Szymański, Deptuła, & Krawczyk, 2013). Concretely, in the approach described in this work, TF-IDF weighting (Salton & McGill, 1986) was applied to the features and similarity between two articles A_i, A_j was computed according to Formula 2:

$$s(A_i, A_j) = \frac{\|a_i \odot b_{ij}\|_1 + \|a_j \odot b_{ij}\|_1}{\|a_i\|_1 + \|a_j\|_1} \tag{2}$$

where a_i and a_j are vectors representing articles A_i and A_j, respectively, and b_{ij} is a vector defined as follows:

$$b_{ij}[f] = \begin{cases} 1 & \text{when } a_i[f] \neq 0 \ \wedge \ a_j[f] \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

[4] http://en.wikipedia.org/wiki/Wikipedia:Category_names

In order to compute the similarity between categories, the categories were expressed as sums of the vectors representing articles assigned to them. For each category C_i, the corresponding vector c_i was computed:

$$c_i = \sum_{A \in C_i} a \tag{4}$$

where a denotes a vector representing an article A belonging to the category C_i. Having categories C_i, C_j expressed as vectors c_i and c_j, respectively, the similarity between categories was computed according to Formula 2.

According to the study described in (Szymański et al., 2013), a similarity metric based on representation of articles using words as features yields good results. A test was performed on data obtained from English Wikipedia. Pairs of the most similar categories were evaluated by assigning each pair a score from 0 to 10. In top 100 pairs of similar categories, 88% were considered reasonable as well (got a score of at least 5).

The similarity between articles, however, can be defined in other ways than using words occurring in documents. Another possibility is to base it on the hyperlinks between pages (Szymański, 2010; Szymański et al., 2013). In such a scenario, it can be assumed that the more links connect two articles A_i and A_j, the more related they are, and therefore the similarity function can be defined in the following way:

$$s(A_i, A_j) = \frac{\left|L(A_i \to A_j)\right| + \left|L(A_j \to A_i)\right|}{|L(A_i)| + \left|L(A_j)\right|} \tag{5}$$

where $|L(A_i \to A_j)|$ is the number of links from article A_i to A_j and $|L(A_i)|$ is the overall number of links in article A_i.

The approaches employing words-based and links-based similarity functions were both implemented in Szymański et al. (2013). It turned out that the metric based on links performed very poorly. Less than 10% of the top 100 discovered pairs of similar articles were actually related pages, and the fraction of strongly related documents in this set was as low as 2%. The tests were run on the data from English Wikipedia, which is hyperlinks-rich. The conclusion drawn from the experiments was that the similarity between articles is not proportional to the number of links. The reason why such a metric yields bad results can be illustrated with outcomes of the Wiki Game[5]. This is an online game where a player is assigned an origin and a goal – usually seemingly unrelated Wikipedia articles – and has to navigate from the origin to the goal by hyperlinks on pages using the smallest number of transitions. There are many examples of unexpectedly short paths between pages on very different subjects. A good example is the hyperlinks path between Wikipedia pages on bacteria Clostridium Difficile and Eastern Orthodox Church: `Clostridium Difficile` ↦ `United Kingdom` ↦ `Eastern Orthodox Church`. The hyperlinks distance between these two completely unrelated subjects turns out to be very short. In the triple, both pairs of directly connected pages are vaguely related. It seems that the more hyperlink-rich a Wikipedia is, the less useful links-based similarity becomes. In the experiments conducted on Simple English Wikipedia, also reported in Szymański et al. (2013), the same similarity function yields much better results. Average number of links per article in Simple Wiki is smaller than in English Wikipedia, yet the links are more meaningful there.

Nastase and Strube (2013) also described an approach to mining relationships between Wikipedia categories which is based on links. However a somewhat more strict policy as to selecting hyperlinks was used. The authors focused on page infoboxes, as these usually gather the most significant information about corresponding subjects.

2.3. Indirect relationships

The methods discussed above concerned direct relationships between items or categories. They can, however, be extended: for example, lets assume that there are n transactions in the database related to ketchup and minced meat and n transactions related to mustard and minced meat. There is no history of a basket containing both ketchup and mustard, therefore no relationship exists between those two items according to the direct metrics. Indirect association algorithms attempt to improve results capturing such relationships.

The IDARM* algorithm proposed in Kazienko (2009) makes use of direct association rules to compute indirect relationships. Both direct and indirect rules are later combined into what is called complex association rules. In the context of web traffic data, the latter are superior to results obtained only using direct rules mining, because the coexistence of pages in one session is constrained by hyperlinks between pages.

In the case of Wikipedia data, indirect similarity can be computed with the use of the existing hierarchic relationships. This idea was proposed by Pang and Biuk-Aghai (2010), where what is called an aggregate similarity is computed in the following three steps:

1. category graph is transformed in such a way that all the loops are eliminated
2. direct category similarities are calculated in the way proposed by Holloway et al. (2007)
3. aggregate similarities are computed as linear combination of direct similarities and descendants' similarities

One of the method's parameters is the number of levels down the category hierarchy considered when retrieving descendants. Because this variable can potentially be greater than one and the category hierarchy contains loops, authors propose to eliminate the loops from the graph. In order to do that, the structure is traversed in breadth-first-search order starting from a top-level category. Any processed node that has already been visited gets removed from the graph. All the nodes in the resulting graph have at most one outbound edge (to their parent), and the top-level category does not have a parent, therefore all the cycles are eliminated. On every level, categories are processed in the order defined by their identifiers in Wikipedia. Since ids reflect the chronological order, i.e. older categories have smaller identifiers than younger ones, in cases when a category has two parents at the same level, an edge connecting with the older parent is retained in the final graph.

The final result, called aggregated similarity between categories, is a linear combination of direct similarity and the similarity between subcategories:

$$s_a(C_i, C_j) = w_1 \cdot s(C_i, C_j) + w_2 \cdot s'(C_i, C_j) \tag{6}$$

where $s(C_i, C_j)$ is a direct similarity between classes C_i, C_j and $s'(C_i, C_j)$ denotes average direct similarity between their descendants. It is required that $w_1 + w_2 = 1$ so that the resulting value is in the same range as direct similarity. Aggregated similarity can be demonstrated using following example: assume there are six classes A, B, C, D, E and F. B and C are subclasses of A, while E and F are children of D. Expression below represents an aggregated similarity between A and D:

$$s_a(A, D) = w_1 \cdot s(A, D) + w_2 \cdot (s(B, E) + s(B, F) + (C, E) + (C, F)) \tag{7}$$

Pang and Biuk-Aghai (2010) report that the best results – in terms of the magnitude of aggregate similarity – were obtained using

[5] http://thewikigame.com/

only first order subcategories and weights values $w_1 = 0.33$ and $w_2 = 0.67$. The concrete results presented in the paper – in terms of similarity values for pairs of categories – are fairly modest. The authors computed similarities between top level categories of Wikipedia (*Science, Geography, Everyday life, People, Religion, Literature, History* and *Knowledge*), while the results included in the paper are restricted to pairs formed by *Science* and one of the other categories. The amount of data presented in Pang and Biuk-Aghai (2010) is certainly not enough to draw any strong conclusions about the proposed metric. Also the results are related to abstract, broad, categories and it is not easy to evaluate them. In the example, it is difficult to intuitively say whether the similarity between *Science* and *Everyday life* should be greater than the similarity between *Science* and *History*.

2.4. Classification of articles to categories

The task of automatic classification of Wikipedia pages – finding relations between articles and categories – is strongly related to learning associations between categories themselves. Approaches can be divided into three categories:

- Graph based – depending on the existing hierarchy of categories and links between pages and categories
- Content based – making use of articles' content features, e.g. words, infoboxes, metadata...
- Network based – making use of hyperlinks between pages.

Farina (2010) describes a graph based solution to a slightly simplified version of the problem, i.e. assigning articles to some selection of macrocategories. This approach relies on the assumption of topic inheritance. This is how the method is explained in Farina (2010): *So, suppose for example that the article "Barack Obama" is labeled with four categories, two of which are assigned to "Politics" and the third one to "Arts", and the remaining one is equally close to "Law" and "People": then the article will be considered related to "Politics" with a score of 0.5, to "Arts" with a score of 0.25 and to "Law" and "People" with a score of 0.125 each.*

A similar method, also based on graph approach, was described by Perez, Feo, West, and Lee (2012), although in this case the stated problem was suggesting categories that might be missing for an article in Wikipedia. The method is based on the assumption that similar articles are assigned to similar categories. In order to find all categories related to a given article *A*, first, similar articles are retrieved – these are the ones that already share at least one category with *A*. Then, all the categories connected with the similar articles are considered. Each of them is assigned a score depending on how many times it occurred. Suggestions for *A* are those categories that received the highest score, i.e. that are the most related to similar articles.

Gantner and Schmidt-Thieme (2009) and Szymański (2010) proposed content-based methods employing a one-vs-all scheme SVM classifier. In Gantner and Schmidt-Thieme (2009) two strategies of selecting classes to labels were tested: RCut (select top t classes) and SCut (select all classes activated above threshold s). Unfortunately, the evaluation of these two approaches was performed using only two classes, which is insufficient to generalize about their actual effectiveness. An other method for creating a large-scale classifier has been proposed in Draszawka and Szymański (2013) where the authors use a modified K-NN classifier adapted to solve multilabel and multiclass classification problem.

Another graph-based method was proposed in Schonhofen (2006). In this case, however, associations between articles and categories are not discovered with machine learning techniques. Instead, for each article the top 20 words were selected according to a TF-IDF weighting. The strength of the relationship between the given article and the category was then estimated as the similarity between category title, and the words selected for the article.

Szymański (2010) describes an article classification algorithm using network-based features. Like in the content-based approach reported in this paper, the TF-IDF weighting was applied to hyperlink features. Obtained classification accuracy was similar as in the case of representing articles by words. Another network model was proposed in Colgrove, Neidert, and Chakoumakos (2011), yet a different representation was assumed there. Considered hyperlink connections were not restricted to direct ones. In order to determine the relationship between an article *A* and a category *C*, eight features were defined expressing the numbers of inbound and outbound connections between *A* and *C* in both direct and indirect cases.

The task of construction of large-scale classifiers has been the subject of many studies. Particularly, the Pascal challenge[6] should be mentioned here, as it set up a competition for building a categorization tool for Wikipedia.

2.5. Wikipedia as a semantic network

Multiple significant projects have been launched over the past decades in order to capture concepts, and relations between them, into machine-readable semantic networks. Perhaps the most notable of them is WordNet (Fellbaum, 1998), which has been under development for almost thirty years. All of WordNet's resources are hand-crafted, thus, all the facts it stores are highly reliable. However, this quality comes at a cost. Its evolution is very slow; particularly it takes time until new concepts are introduced. For example, the recent version of WordNet does not define the concept of crowdsourcing, even though this term has been used for at least ten years (Wired 14.06, 2013).

On the other hand, a wide variety of automatically-created semantic networks are available. ConceptNet (Speer & Havasi, 2012) combines information from other sources, including, but not limited to, Open Mind Common Sense (Singh et al., 2002), WordNet, DBPedia (Lehmann et al., 2013) or the Verbosity game (Luis, Kedia, & Blum, 2006). The primary issue inherent in automatically created networks is noise.

Wikipedia is a large scale repository, created mostly by human editors. It is also dynamic – new concepts are usually described in Wikipedia very fast, and existing content is kept up to date (Holloway et al., 2007). Therefore, its resources are in some way intermediate between hand-crafted and automatic approaches. DBPedia represents each Wikipedia article as a concept (Lehmann et al., 2013). Advanced methods of extracting structured information from the wiki code were applied. For example, DBPedia makes use of infoboxes, or attempts to extract coordinate data from articles. Yago2 (Hoffart et al., 2009) uses similar techniques, but it is also based on relations stored in WordNet. Both DBPedia and Yago2 support multilingual lexicalisations of concepts.

Nastase and Strube (2013) described an approach that was used to create the WikiNet semantic network. Initially WikiNet was crafted from categories, articles and relationships that are explicitly stated in Wikipedia. This network then evolved with the discovery of new links based on matches in category names. Additional relations were also obtained with the method, which makes use of infobox hyperlinks. Both these approaches have been described in Section 2.2. Similarly, as in DBPedia and Yago2, redirect, disambiguation and cross–language links existing in Wikipedia were used to create multilingual lexicalisations of the concepts.

WiseNet is another project aiming to build a Wikipedia based semantic network (Moro & Navigli, 2012). In this approach the

[6] Pascal Challenge in Large Scale Hierarchical Text Classification http://lshtc.iit.demokritos.gr

authors attempt to extract associations between concepts from article content. Concretely, sentences contained in pages are checked against patterns allowing one to coin a text into a relationship. For example, in the sentence: `Natural Language Processing is a field of computer science`, the left and right parts can be recognized as Wikipedia pages *Natural Language Processing*, and *Computer Science*, and therefore the relation *is a field of* can be stated between those two concepts.

3. Category similarity metrics

To calculate associations between Wikipedia categories we use five metrics. The first two, described in the Sections 3.1 and 3.2, are built upon ideas mentioned in previous sections (Holloway et al., 2007; Szymański et al., 2013). Their description, as it was used in our research has been given in this section. The others, which make use either of a model or predictions done by multilabel text classifier, are our contribution.

3.1. (A) Metric based on assignment of articles to categories

Similarity measure (A) is a baseline approach proposed by Holloway et al. (2007). The method is based solely on article labels, and the direct similarity is computed according to Formula 1.

3.2. (B) Similarity between categories represented by words

Metric (B) is a content-based method. According to Szymański et al. (2013), this approach performs well when documents are represented in a Vector Space Model (VSM) using words as features and with a TF-IDF weighting. However, instead of the Formula 2 in metric (B) we use the cosine similarity function defined by Formula 8 since it is invariant to the lengths of the vectors. It should be noticed that the other variants of VSM can be also used here. A similar approach, based on different representation has been shown in Chernov, Iofciu, Nejdl, and Zhou (2006). The method instead of words employs the links between categories. The other option is shown in Szymański et al. (2013) usage of links between articles.

$$\left(A_i, A_j\right) = \frac{a_i \cdot a_j}{\|a_i\| \cdot \|a_j\|} \tag{8}$$

where a_i is a vector representing document A_i.

3.3. (C) Classifier coassignments based metric

Similarity measure (C) makes use of predictions made by a multilabel text classifier. The underlying idea is that when the classifier predicts an object to be an instance of both class C_i and C_j, it is a premise that those classes are alike. Cases when the classifier predicts only one of these categories imply the opposite, i.e. that C_i and C_j are different. This measure is very similar to (A), however, computations are performed using predicted labels instead of the original ones. Putting it formally, similarity between classes C_i and C_j can be computed according to Formula 1, yet in this case $A_k C_i$ equals 1 when the classifier predicted class C_i for article A_k and 0 otherwise.

There is one additional constraint that had to be introduced to the metric (C), related to the number of occurrences of a particular class in the predicted labels. Concretely, categories assigned to less than three test objects are ignored. In such cases the information about the class is insufficient to make any assumptions about its similarity to other classes. For example, if C_i occurs only in one label predicted for object A and C_j was assigned to four inputs A_1, A_2, A_3, A_4, then similarity between C_i and C_j would be as high as 0.5. Tests on real data show that going ahead with such vague premises brings a lot of noise to the results. Eliminating underrepresented classes – like C_i in the above case – causes the metric quality to improve significantly.

Table 1
Possible pairs of original and predicted labels with regard to classes C_i and C_j.

Original label		Predicted label		Impact on C_i, C_j similarity
C_i	C_j	C_i	C_j	
0	0	0	0	–
0	0	0	1	–
0	0	1	1	–
0	0	1	0	–
0	1	1	0	Similar
0	1	1	1	Similar
0	1	0	1	Different
0	1	0	0	Different
1	1	0	0	–
1	1	0	1	–
1	1	1	1	–
1	1	1	0	–
1	0	1	0	Different
1	0	1	1	Similar
1	0	0	1	Similar
1	0	0	0	Different

3.4. (D) Classifier misspredictions based metric

Metric (D) also makes use of predictions made by a classifier, although in this case the original labels are taken into account as well. Specifically, it is assumed that when the classifier predicts class C_j for an instance of C_i, it is a premise that these categories are similar. Table 1 presents all possible combinations of actual and predicted labels with regard to classes C_i and C_j. Here, value 1 denotes occurrence of a corresponding class in the label, and 0 means the opposite. Note that every label can contain other categories than C_i or C_j, yet it is of no importance when considering similarity between C_i and C_j.

The last column of Table 1 contains interpretations of the corresponding cases. The following two assumptions provide an explanation FOR these interpretations:

- When an original label contains both or neither of C_i and C_j, prediction gives no information on their relation
- When a category is present in the original label, it does not matter for the similarity if that category was correctly recognized by the classifier and put into the predicted label

Formula 9 expresses the above idea as a similarity function:

$$s\left(C_i, C_j\right) = \frac{\left|C_i \wedge \sim C_j \rightarrow C_j\right| + \left|C_j \wedge \sim C_i \rightarrow C_i\right|}{\left|C_i \wedge \sim C_j \rightarrow *\right| + \left|C_j \wedge \sim C_i \rightarrow *\right|} \tag{9}$$

where $|A \rightarrow B|$ denotes the number of objects such that the original label satisfies condition A, and the predicted label meets condition B. C_i denotes occurrence of the class C_i in the label, while $\sim C_i$ means category C_i is not included in the label. The star symbol $(*)$ denotes no condition.

3.5. (E) Metric based on angles between hyperplanes

The metric (E) is also based on a classifier; however, contrary to the previous two metrics, in this approach similarities between the classes are inferred from a trained model rather than labels. Specifically, in this case it is assumed that the classifier's linear and multiclass functionality was achieved using a one-vs-all scheme. In such a scenario, the model consists of as many hyperplanes as

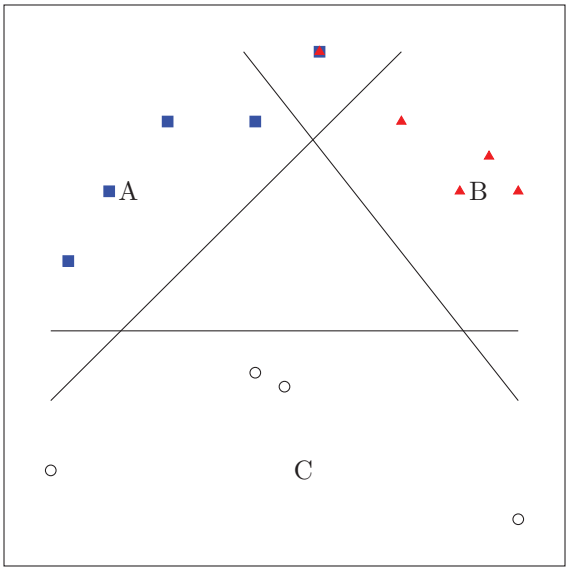

Fig. 1. Toy example of a linear classifier with one-vs-all scheme.

there are classes, each hyperplane separating instances of a corresponding class from other objects.

Fig. 1 illustrates such a classifier in a simple 2D scenario. In our approach we employ a linear classifier as our data are high dimensional and sparse. It is known (Joachims, 1998), linear classification is usually sufficient in most of text classification problems. In the case of other data, or if there is requirement of the higher classifier accuracy, we may use other, non-linear kernels. It it must be stressed here that the, used by us, linear classification has several advantages: it is fast, we only need to optimize the C regularization parameter.

In the example presented in Fig. 1 three classes are denoted by squares (A), triangles (B) and circles (C). Note, that this is a multilabel case and one of the objects is assigned to two classes (triangle over a square at the very top). Straight lines denote decision boundaries.

Only looking at the straight lines (and having in mind which side of a line is *positive*), it is possible to say which classes are the most similar. In the Fig. 1, the angles between lines corresponding to classes A and B are roughly right angles, while other angles (A-C and B-C) are about 135°. Thus it can be inferred that the similarity between categories A and B is bigger than in the other cases. This finding aligns well with graphical interpretation of the dataset presented above: squares and triangles are relatively close to each other, and circles are located away from the rest of objects, therefore the similarity between class A and B seems greater than in any other pair.

However, experiments have shown that the above approach performs very badly on real data. In some cases, angles between hyperplanes pertaining to sorely different classes were very small, sometimes the smallest angle value was related to a pair of categories that were not similar at all. It was found that the angle values were strongly dominated by compliance over *negative* features. Each hyperplane is defined by its normal vector n. Feature f is positively correlated with a corresponding class when $n[f] > 0$, while $n[f]$ it indicates negative correlation. Inspection of vectors n has shown that, in the case of text classifiers with a large number of classes, the proportion between positively and negatively correlated features is extremely skewed. Typically, out of hundreds of thousands of features, only up to a few hundreds were positively correlated. As a result, in such very high dimensional space, the angle between two hyperplanes may be very small, as those hyperplanes align very well over their numerous common *negative* dimensions. In order to fix this issue, for each vector n a corresponding vector n' is defined:

$$n'[f] = \max(n[f], 0) \tag{10}$$

Given classes C_i, C_j and provided that n_i and n_j are normals to their respective hyperplanes trained according to a one-vs-all scheme, similarity between the categories can be expressed in the following way:

$$s(C_i, C_j) = \frac{n_i' \cdot n_j'}{\|n_i'\| \cdot \|n_j'\|} \tag{11}$$

3.6. Metric aggregation algorithm

The metric aggregation algorithm used in our approach is an adaptation of the approach described by Pang and Biuk-Aghai (2010). Weight of similarity between two categories is given by Formula 6 that is a linear combination of direct similarity and the similarity between subcategories.

One of the parameters considered by the authors was the number of levels down the category tree used to compute the aggregated similarity. As reported in the paper, using only the first level, i.e. immediate children of the category, guarantees obtaining optimal results. In such a setting, the step of transforming the category graph into a tree is not necessary, and therefore can be dropped. It also seems beneficial to skip the category graph transformation, since the loop elimination algorithm removes roughly half of the graph's edges (in English Wikipedia the number of edges between nodes in the graph is over twice the number of categories).

Pang and Biuk-Aghai (2010) also considered parameters w_1 and w_2, which define the proportion of direct similarity and average direct similarity computed between descendants. According to the presented results, values $w_1 = 0.33$ and $w_2 = 0.67$ are optimal, i.e. they maximize the magnitude of obtained similarities. Exactly these values of w_1 and w_2 will be used in our work.

4. Experiments

4.1. Data preparation

The data used across all of the experiments was obtained from the English Wikipedia dump generated on the 4th of February 2013. Wikipedia dumps, comprising a variety of .xml and .sql files, have a fairly complex structure. Therefore, a tool named MATRIX'u (Szymański, 2013, 2014) was used to transform the Wikipedia dumps into a form that is more handy to process and compute. Among its other functions, MATRIX'u allows one to define a set of categories in the dump, select the required representation method for them and generate data files according to that representation. In this case, all Wikipedia categories – namely everything descending from *Fundamental categories* – were included in the final data set. The articles were represented in the Vector Space Model (Salton et al., 1975) with stemmed words as terms and using TF-IDF weighting (Salton & McGill, 1986).

An analysis of the dataset produced by MATRIX'u unveiled that further processing was required before the experiments could be run. First, it contained a number of Wikipedia administrative categories[7], for example *stubs* or *templates*, useless in the context of this work. It also contained categories that grouped vaguely connected pages. For example, there are hundreds of categories in Wikipedia related to years from history, one of them being *1989*.

[7] http://en.wikipedia.org/wiki/Wikipedia:Categorization

Helpful as they are when it comes to searching Wikipedia for certain events, these categories are fairly inconsistent and therefore have little value in category metric experiments. Another group of categories in the dataset that required some action were those containing very few articles. More then half of the 850,000 original categories were assigned to less than five pages. Since four or less articles is most likely not enough data to generalize to the whole category, those categories of small cardinality had to be handled somehow. Finally, the Wikipedia category system is fine-grained. Large categories are subject to diffusion, thus a vast amount of articles are assigned to overly specific categories, while more general ones remain relatively modest in size. For example, the category *Economy of Poland* is assigned to only 16 pages and the vast majority of descriptive content is diffused into specific subcategories.

In order to adress the above issues, and reduce the dataset's significant size, the following steps were undertaken:

- Granularity reduction
- Small categories removal
- Category by name–filtering

The remainder of this section describes those steps in detail.

4.1.1. Granularity reduction

The purpose of granularity reduction was to expand general categories while removing the very specific ones. This was performed by reverting the diffusion process that Wikipedia editors apply to large categories. Algorithm 1 presents how the granularity reduction was performed.

Algorithm 1: Granularity reduction

```
Data: C – the set of all categories
Data: L_min – minimum leaf category size
repeat
    L ⇐ c ∈ C : c is a leaf;
    R ⇐ l ∈ L : size(articles(l)) < L_min;
    forall the r ∈ R do
        forall the p ∈ parents(r) do
            forall the a ∈ articles(r) do
                add a to articles(p);
            end
        end
        remove r from C;
    end
until size(R) > 0;
```

The procedure has a single parameter $L_{\min}$ denoting minimum leaf size. In an iterative process, all the leaves assigned to less than $L_{\min}$ pages are merged with their parent categories. This way, the process of dividing big categories performed by Wikipedia editors, is reverted. As a result, the overall number of categories in the dataset drastically decreases. The number of low cardinality nodes in the category structure lowers as well, and the output categories are more coarse-grained.

4.1.2. Small nodes removal

The granularity reduction algorithm partially solves the problem of categories containing small number of pages by iteratively merging small leaves with their parents. However, some non-leaf categories having only a few pages remain in the structure. For example, the category *Poland* contains two articles after the granularity reduction, and also multiple fair-sized subcategories on specific topics related to Poland. Such categories are removed from the graph, formally any category containing less than $N_{\min}$ pages is eliminated. Fig. 2 presents an example category graph, and Fig. 3 shows the output structure of the granularity reduction algorithm and small nodes removal procedure.

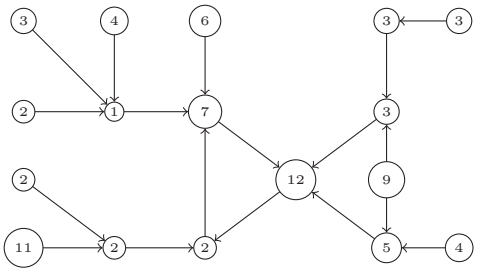

Fig. 2. Sample directed category graph. Nodes represent categories, edges show hierarchical relationships between categories. Values inside the nodes denote categories' article–wise cardinality.

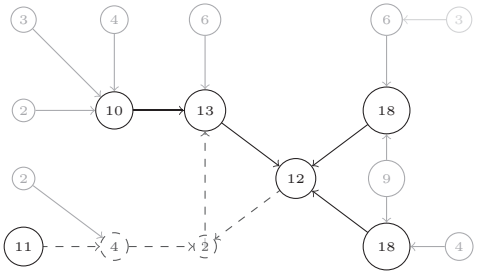

Fig. 3. Category graph from Fig. 2 transformed with parameters $L_{\min} = 10$ and $N_{\min} = 5$. Leaves merged with their parents are faded out; dashed stroke denotes the nodes removed due to having less than $N_{\min}$ articles assigned.

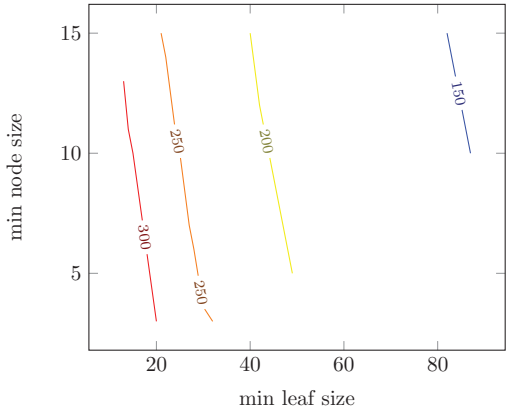

Fig. 4. Number of categories in the dataset depending on parameters $L_{\min}$ and $N_{\min}$ assumed for granularity reduction and small nodes removal.

In this example the number of categories dropped from 17 to 6, while only (at most) six articles were removed. Note that removing a category does not necessarily remove articles from the data set since some of those might belong to other categories. Because an article can be assigned to more than one category, the granularity reduction boosts parent classes' sizes very efficiently. In the above example, the category containing nine pages (in the bottom–right of the graph) contributes to the sizes of two other classes. Since the granularity reduction is an iterative process, the number of assignments of an article propagating up the category structure grows exponentially with succeeding iterations.

The granularity reduction and small nodes removal parameters are strongly related. It is not enough to investigate their influence on the overall categories number separately in order to decide what values to choose for $L_{\min}$ and $N_{\min}$. A contour plot in Fig. 4 shows the relation between $L_{\min}$, $N_{\min}$ and the number of categories in the resulting dataset.

Table 2
Sample patterns used for category filtering.

stub[_s\Z]	.*_language_films.*
(_—\A)articles(_—\Z)	_by_year
template	_shot_in
wikipedia\Z	female_
(_—\A)disambiguation(_—\Z)	male_
(_—\A)(deaths—births)(_—\Z)	with_proper_names
(_—\A)\d+s?(_BC)?(_—\Z)	people_who
years	Albums_with_cover_art_by_
(_—\A)century(_—\Z)	speaking_countries
(_—\A)(millennia—millennium)(_—\Z)	alumni
(_—\A)(unknown—uncertain)(_—\Z)	_screenwriters

Table 3
Approximate numbers of categories, articles and features in the final dataset that was used in the experiments.

# Categories	# Articles	# Features
145,000 (850,000)	3,800,000 (4,000,000)	3,000,000 (4,000,000)

The contour lines on the plot are almost at right angles to the $L_{\min}$ axis, therefore the number of categories in the output dataset is much more sensitive to $L_{\min}$ than to $N_{\min}$. Finally, the dataset generated by MATRIX'u was processed with parameters $L_{\min} = 40$ and $N_{\min} = 12$.

4.1.3. Filtering categories by name

In the last phase of data preparation, the categories are filtered according to their names. In this step we eliminate the aforementioned Wikipedia administrative categories, and categories grouping vaguely-related pages. The filtering is performed by testing the name of every category against each of the predefined regular expressions. A category whose name matches any of the expressions is filtered out of the dataset. Table 2 presents a sample subset of the patterns used for the filtering.

Category filtering eliminates about 60,000 categories. Table 3 displays properties of the final dataset, which was obtained by applying granularity reduction, small nodes removal and category name filtering to the data generated by MATRIX'u. Numbers in parentheses refer to the original dataset generated by MATRIX'u application from the full English Wikipedia.

4.1.4. Feature-wise small values truncation

The dataset obtained through the process of filtering was a sparse matrix containing $5.5 \cdot 10^8$ non-zero elements. It was too large to fit the memory of a machine equipped with 8 GB RAM, causing a variety of issues, for example with training a classifier in a one-vs-all scheme. Thus further processing was performed in order to shrink the dataset's representation. The dataset contained numerous values that were very close to zero. Removing (actually replacing with zeroes) all elements less than some arbitrary number would cause an uncontrolled loss of information, because features in the dataset had different ranges, therefore the threshold was established individually for each feature. Concretely, values in every column were scaled to the range [0, 1], and all the elements that were less than 0.02 after the scaling were removed. In order to preserve the original weighting and ranges of the features, the scaled values were only used for the purpose of deciding whether to keep an element in the dataset or not; whenever an element was above the threshold, its original value was retained in the output dataset. After the truncation, the resulting matrix contained $2.3 \cdot 10^8$ elements.

4.2. Multilabel text classifier

Certain category metrics described in this paper infer similarities between classes from predictions made by a classifier. Thus, an efficient multilabel text classifier implementation is one of the core aspects of this work. It was realized using a state–of–the–art linear Support Vector Machines (SVM) (Hearst et al., 1998) library named LIBLINEAR (Fan, Chang, Hsieh, Wang, & Lin, 2008), a multiclass classifier that scales well to problems containing large numbers of training examples. Since LIBLINEAR itself does not support multilabel scenarios, and also in order to gain full control over the label prediction details, LIBLINEAR was employed only as a binary classifier. This decision was also bolstered by the claims presented by Rifkin and Klautau (2004) that a multiclass classifier's quality depends heavily on the underlying binary solvers' performance, while the selection of the multiclass scheme is of minor importance. Several approaches to multiclass problems are known, and have been well studied in the literature (Hsu & Lin, 2002; Kai Bo Duan & S. Sathiya Keerthi, 2005; Rifkin & Klautau, 2004). They vary from simple one-vs-all and one-vs-one strategies, to advanced solutions comprising only a single optimization problem, for example the algorithm presented by Crammer and Singer (2002). Given that applying advanced multiclass schemes results in moderate accuracy payoffs, and in order to keep the scope of this project reasonable, the one-vs-all strategy was implemented in the classifier.

Early prototypes of the classifier were implemented in Python, although their performance was not satisfactory due to the communication overhead – it was necessary to pass data to LIBLINEAR binaries through files. The final version was implemented in C++, which allows us to call core LIBSVM routines directly, making use of very fast shared memory. LIBSVM itself is a cross-platform library. However, due to the way parallel processing was realized, the implemented classifier requires POSIX API, which practically restricts platforms to UNIX-based only.

5. Metrics evaluation and discussion

Previous sections describe how five similarity metrics (A)–(E) were formulated, implemented and computed. Pang and Biuk-Aghai (2010) proposes an algorithm of direct similarity aggregation: a way to transform one similarity metric into another using hierarchical relationships between classes. However, it is not clear how the aggregation step influences quality of the results. In order to address this uncertainty, each of the metrics was computed in two versions: direct, according to the formulation from Section 3, and aggregated, where new values depend on direct similarity and the original Wikipedia relations. As a result, a total of ten metrics were obtained. This section describes an attempt to evaluate and compare them.

Even for humans, assessing a similarity value between two categories is not an easy task. What is more, perhaps there is no such thing as objective similarity: every person can have their unique view on how similar two concepts are, depending on their culture, education and experience. Either way, because of the multitude of possible category pairs – over 21 billions for the dataset considered in this research – it is practically impossible to evaluate a metric by hand, unless via some extremely successful massive online collaboration project like ReCaptcha (Von Ahn, Maurer, McMillen, Abraham, & Blum, 2008) or Duolingo[8]. Also an automatic metric evaluation is out of question – otherwise correct values of similarity could be computed in an optimization process, for example with use of a genetic algorithm (Goldberg, 1989).

[8] http://www.duolingo.com/#/info

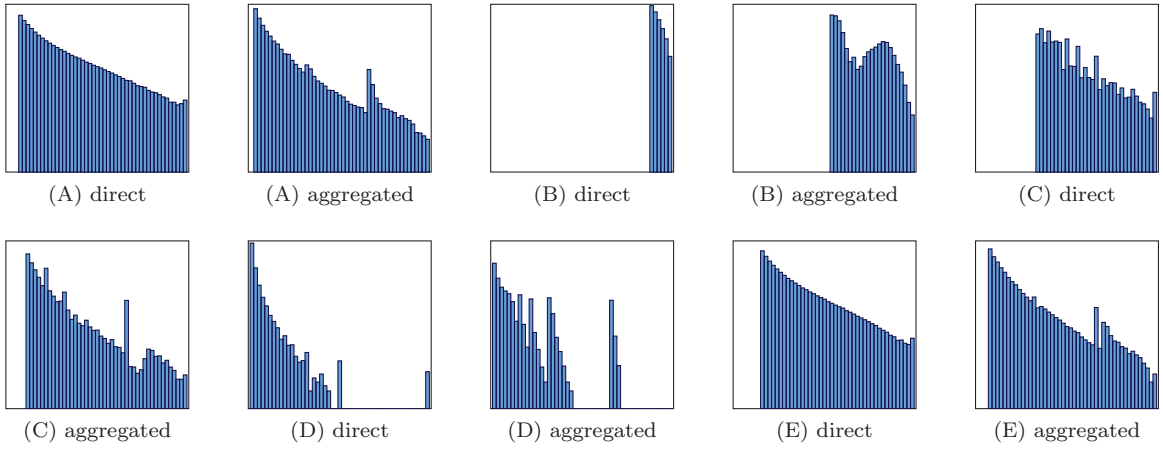

Fig. 5. Histogram presenting distribution of similarity values in top 1,000,000 pairs for each of the metric. Axes' ranges are fixed across the plots to [0, 1] on *x* axis and [1, 500000] on *y*. Axis *y* is logarithmically scaled.

Even though a thorough evaluation of each metric is unrealistic, it is possible to learn some of its properties by a fragmentary inspection. Concretely, the approach applied here involves examination of pairs of categories on two "levels": *top* – pairs of categories that are the most similar according to a particular metric – and *deep* – pairs 50,000 down from the top. This way, by assessing only a very small fraction of pairs, enough evidence is gathered to estimate the metric's performance.

Another way to evaluate metrics' quality is to use original Wikipedia categories as a reference set. Those original categories are mostly taxonomic, which means that they join very related articles. Measurement of how well a metric restores them can therefore be very informative: very few restored intercategorial links perhaps mean that the metric does not perform very well. On the other hand, a metric that only restores existing relationships, and does not discover new ones, has a good quality, but is not very useful.

Evaluation of each of the similarity measures is not the only way to compare them. Calculating similarity between metrics also sheds more light on the results, as does plotting distributions of similarity weights generated by metrics. In the next subsections we present results of experiments performed in order to compare the metrics.

5.0.1. Similarity values distribution

One of the tasks performed in order to compare the metrics was to plot distributions of similarity weights. Fig. 5 presents those distributions related to each of the metrics. Each metric's top 1,000,000 values were used to generate the plots. In every case the axes' ranges and scale are identical in the Figure.

Note that logarithmic scaling was applied to the *y* axis; therefore, in most cases the number of category pairs grows exponentially as the value of similarity decreases. The plots show that the metrics generate similarity values from various ranges. Interpretation of one similarity value therefore must strongly depend on the metric. For example, value about 0.9 denotes a very strong similarity in case of metric (A), but only a little similarity for the metric (B).

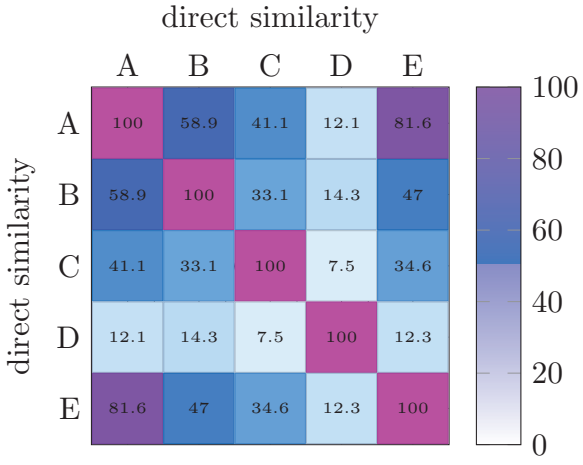

Fig. 6. Overlap of direct similarity metrics measured within top 1000 pairs.

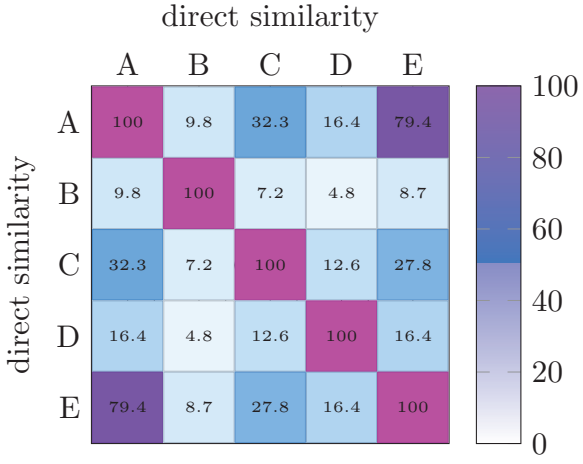

Fig. 7. Overlap of direct similarity metrics measured within top 50,000 pairs.

5.1. Overlap

According to Fig. 5, approaches (A) and (E) have very similar weight distributions, while (A) and (B) are very different. Yet the results from the above section are insufficient to tell how similar (or dissimilar) those two metrics actually are. In order to shed more light on how individual metrics relate to each other, and also how the aggregation step influences the top results of a measure, an overlap was computed between every two metrics. The experiment was performed twice: first, sets of top 1,000 pairs were considered, then an intersection was computed for sets' size equal to 50,000. Figs. 6–11 present similarities between the metrics measured this way. Values on the graphs denote percentage of pairs

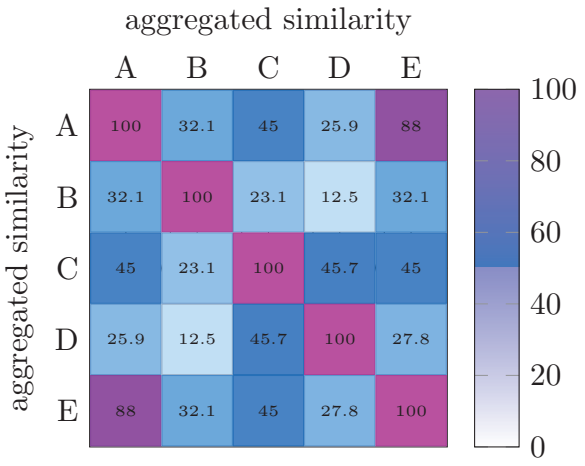

Fig. 8. Overlap of aggregated similarity metrics measured within top 1000 pairs.

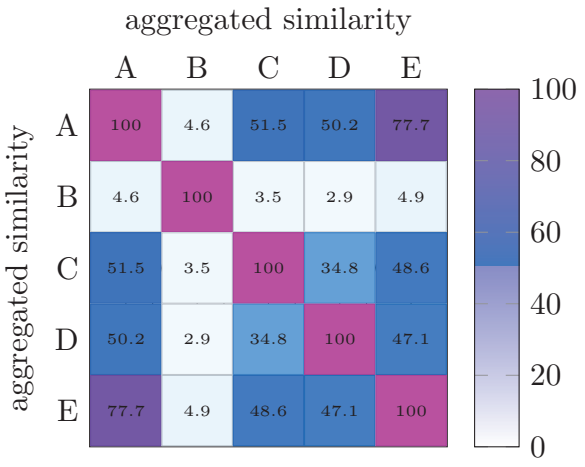

Fig. 9. Overlap of aggregated similarity metrics measured within top 50,000 pairs.

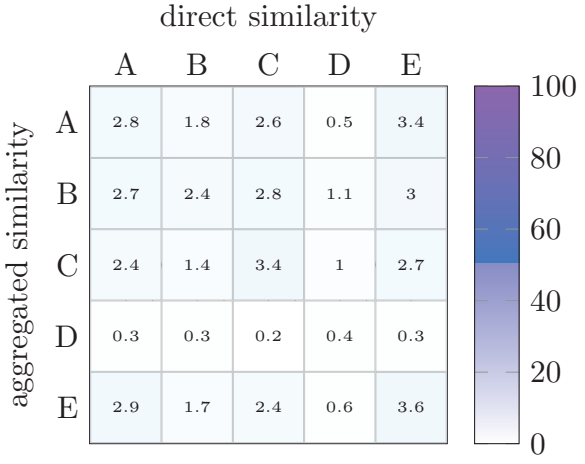

Fig. 10. Overlap of direct and aggregated similarity metrics measured within top 1000 pairs.

shared by sets corresponding to rows and columns. Results obtained for sets of 1000 pairs are shown on the left-hand side of the page, those pertaining to the top 50,000 are on the right. Note that the first four matrices are symmetric while it does not hold true for Figs. 10 and 11, where overlap of direct and aggregated similarity measures is considered.

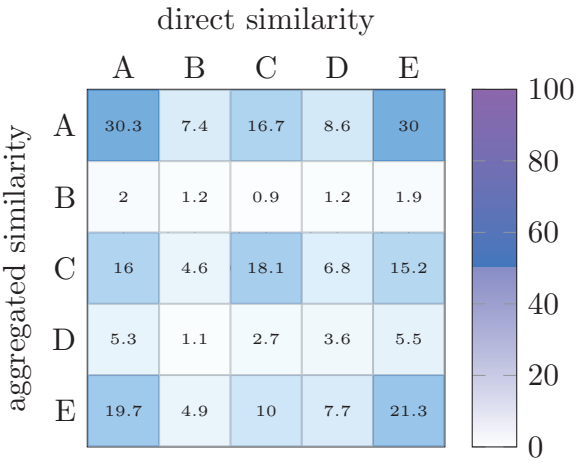

Fig. 11. Overlap of direct and aggregated similarity metrics measured within top 50,000 pairs.

Metrics similarity

Metrics (A) and (E) turned out to be the most similar among all the combinations, both in terms of similarity values distribution, and overlap. Regardless of the size of the sets of pairs, overlap between those two metrics keeps at the level of about 80%. This result is surprisingly high given that (A) and (E) are defined in completely different ways, based on different information about categories. Additionally, the overlap of direct (E) and aggregated (A) on Fig. 11 equals 30%, which is as much as the intersection between direct and aggregated versions of (A), and even more than in the case of two variants of (E) on the same plot. These facts confirm that metrics (A) and (E) have very much in common.

On the other hand, similarity measures (B) and (D) are the most distinctive. There are clearly visible faint crosses in Figs, 7 and 9 resulting from an uncommonly low overlap between (B) and others. Similarly, such a pale cross also occurred in the case of metric (D) for direct similarity in Figs. 6 and 7; however, it saturates on graphs related to aggregated similarity. According to this study, metric (C) was average – it did not distinguish itself as particularly similar or dissimilar to others.

The aggregation algorithm

Results suggest that the aggregation algorithm has a vast influence on category pairs yielded by the metrics. Matrix diagonals in Figs. 10 and 11 present the overlap of direct and aggregated versions of individual metrics. Aggregating results alters almost all the top pairs in the case when sets of size 1000 are considered. Overlap grows when extending those sets to 50,000, yet it still remains small – 30% intersection between direct and aggregated (A) is the highest value.

Another observation is that the aggregation step makes metrics alike. Fig. 9 is clearly darker than Fig. 7. Speaking in numbers, average overlap (computed over fields above the diagonal) is 21.5% for direct, and 32.6% for aggregated, similarity in these figures.

5.2. Restored hierarchical relationships

One of the tested properties of the created metrics was how the original Wikipedia relations were represented in computed pairs of similar categories. These hierarchical relations can be considered strong; therefore, a very low share in rediscovered relations might indicate that a corresponding metric does not perform well. For each measure, pairs of similar categories were sorted by similarity value in descending order. Then, the top 50,000 relationships – pairs with the strongest connections – were considered. A fraction

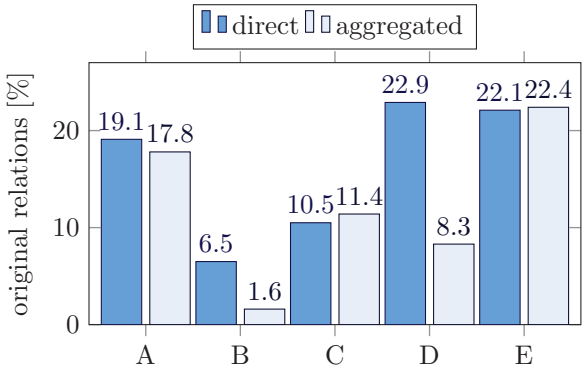

Fig. 12. Fraction of original relations in top 50,000 pairs of each metric.

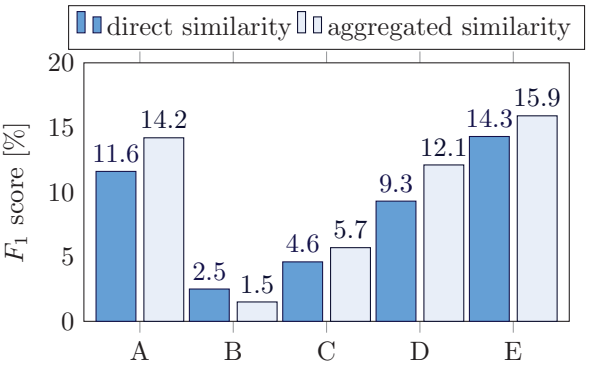

Fig. 13. Comparison of peak F_1 scores for direct and aggregated similarity metrics

of the original hierarchic Wikipedia relationships was computed in each of these sets.

In half of the cases, original relations' share in the top 50,000 pairs was around 20%. In the case of metric (B), those hierarchic relationships were least represented: 6.5% corresponds to 3250, and 1.6% only to 800 of the top 50,000 pairs. These values are suspiciously low, given that the total size of the original Wikipedia relations population is 320,000.

In the case of metrics (A), (C) and (E), the fraction of the original relationships in the strongest pairs sets almost did not change after the aggregation step. The largest drop was observed for metric (D) – from the highest fraction (22.9%) for direct similarity to the third lowest value of 8.3%. The aggregated results have been shown in Fig. 12.

Another way to measure how well the metrics align with the original relationships is to define the problem as restoring those relations. Given a cut-off threshold (minimum similarity value required for a pair of categories to be considered related), precision can be defined as the fraction of restored Wikipedia relations to all discovered relations. Similarly, recall can be defined as the fraction of restored relations to all those existing in Wikipedia. F_1 score equals the harmonic mean of precision and recall. For each metric, the cut-off threshold was selected in a way maximizing F_1. Fig. 13 presents peak F_1 scores obtained by the metrics. Results obtained this way lead to similar conclusions to those that were drawn after considering the restoration in the top 50,000 pairs. The main difference is related to performance of metric (D).

5.3. Manual evaluation

Results presented so far mainly provide information on relationships between metrics, while quality of the implemented similarity measures is still mostly a missing factor. In order to evaluate the quality, manual assessment was performed. Two groups of category pairs were examined to this end: *top* and *deep*. *Top* denotes 100 pairs of categories with the highest similarity according to each metric. *Deep* is also a set of 100 relationships, yet these are sourced from the end of the first 50,000 pairs. So, *deep* elements were chosen in the following way:

```
head relations.data -n50000 | tail -n1000 | sort -R | head -n100
```

where relations.data is a file containing pairs discovered by a given metric and ordered by similarity in descending order. Randomness was introduced because sometimes very similar pairs were grouped together, and shuffling guaranteed topic diversity.

This way, a total of 2000 pairs were selected. All of them were merged and graded collectively, which assured two qualities. Firstly, the person performing the grading could not know which metric was the source of a given pair of categories. This condition made the evaluation fair – otherwise the grader, being unconsciously biased for or against a certain metric, could assert partial results. Secondly, putting all pairs together removed duplicates, so that there were less pairs to grade, and one pair yielded by multiple metrics received consistent grades. All together, the set of pairs to grade contained 1616 elements: 616 from *top* sets and 1000 from *deep* (there were no duplicates across *deep* sets).

Each of the category pairs received an integer grade from range [0, 3]. 0 denotes no relationship between categories, 1 – common domain, no actual relationship, 2 – categories similar, yet relationship not significant 3 – strong relationship.

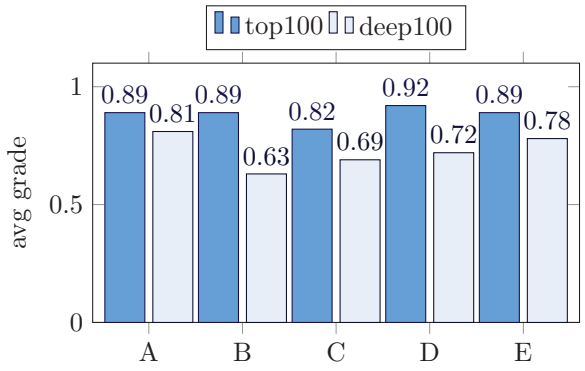

Fig. 14. Average grade for direct similarity metrics.

Fig. 14 presents results obtained by direct similarity metrics normalized to [0, 1]. Each of the metrics was graded higher in *top* sets than in *deep*. Results indicate that similarity measures have an alike quality when measured in top 100 pairs: all metrics apart from (C) scored between 0.89 and 0.92. However, average grades diverge when evaluated 50,000 down the results list. These grades are more informative: average score measured on *top* only sheds light on the top 100 pairs. On the other hand, evaluation of 100 pairs sourced 50,000 entries lower can be a base to draw some conclusions about metric performance. Following this assumption, it is worth noticing that in *deep* sets, metric (A) turned out to score the highest average of 0.81, followed by 0.78 of (E). Therefore these two metrics can be regarded as the best ones considering direct similarity.

Fig. 15 shows average grades asserted for aggregate similarity metrics. Looking at the top 100 pairs, quality degrades when compared to direct similarity. On the other hand, there are only subtle differences between the quality of direct and aggregate measures tested *deep*. The only exception to this rule is metric (B) for which significant degradation was observed.

5.4. Experimental outcomes

Results of the experiments presented in this paper are sufficient to draw conclusions about the aggregation algorithm and the five implemented similarity metrics.

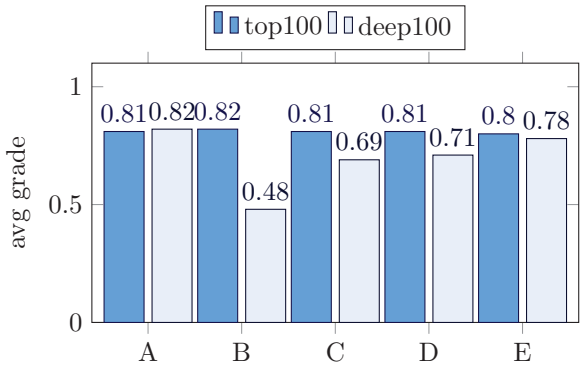

Fig. 15. Average grade for aggregated similarity metrics.

Table 4
Comparison of the metrics: summary.

	Quality	Computation cost	Overall
A	Good	• Easy and cheap • Requires only labels	Very good
E	Good	• Considerable workload • Expensive (single training) • Requires articles representation	Good
D	Middle	• Considerable workload • Very expensive (training k times) • Requires articles representation • Requires prediction	Middle
C	Middle	• Considerable workload • Very expensive (training k times) • Requires articles representation • Requires prediction	Middle
B	Poor	• Fair workload • Cheap • Requires articles representation	Poor

According to the experiments involving overlap of metrics, aggregating similarity vastly rearranges results yielded by direct measures. Looking at quality, it seems that the aggregation step has a slightly negative influence on the top pairs. On the other hand, results measured deep are very similar. It is also worth noting that peak F_1 scores were somewhat higher for aggregated similarities than for direct ones. It seems that aggregation neither improves or deteriorates the results. Mostly, it makes them different.

Essential features of the similarity metrics are presented in Table 4. Metrics are ordered from the best to the worst one.

Experiments unveiled that metrics (A) and (E) were of the highest quality amongst the considered approaches. What is more, it turned out that these two are extremely alike. This fact may come as a bit of a surprise since (A) and (E) are computed in completely different ways. The former similarity measure is based exclusively on labels assigned to Wikipedia pages; the latter makes use of labels as well, but also required representing articles as vectors of features and involves training a classifier and analyzing its model. These are ultimately different procedures, and yet the overlap measured between top 50,000 of each metric's top pairs is as high as 80%.

In order to explain this phenomenon, it is handy to consider what kind of categories can score very high according to metric (A). To be assigned a similarity value of 1, two categories must contain exactly the same articles. Within discovered associations we found pairs that were scored 1, for example the categories *Songs written by James Hetfield* and *Songs written by Lars Ulrich*. Indeed, according to Wikipedia, as of June 2013 each song written by James Hetfield was also written by Lars Ulrich, and the other way around. Therefore, 100% similarity between these two categories is perfectly justified.

Moving on to the metric (E), it is easy to imagine why those two categories should score high as well. Since they contain the same pages, training sets of corresponding binary classifiers were the same, too. Therefore almost identical hyperplanes must have been fitted by the SVM and, as a result, the two categories were assigned almost identical features. Had there been small differences in what articles were assigned to those categories, training sets would still have been alike, and also similar features would have been selected.

To some extent metric (E) follows the logic of (A), yet it does it in a much more complicated process. Complexity is not an advantage; however, it introduces some additional potential to (E). The ability to detect similarities in texts of different articles, and not only to rely on shared pages. The complexity of the approach based on classifier is its important issue. Comparing to the approaches that perform implicit analysis of two sets, the method is slow, and due to the requirement of constructing the classifier it can be used in off-line processing. The complexity of the metrics computation based on mining direct relationships (A,B) that compare each category with another is $O(k^2)$, where k denotes the number of the categories. While we use the SVM classifier we have complexity $O(k{*}d^2)$ where d denotes size of the dataset. Depending on the metric, the classifier training complexity is multiplied that is selected in the Table 4. It should be noticed that in practical applications a big problem can be storing the hyperplanes that, after training SVM, are represented as dense vectors and occupies a large disk space. Also, despite linear complexity, multiple lookups into this structure can cause significant delays, so application of the dedicated indexes (eg. similar to those proposed in Yu, Ooi, Tan, & Jagadish (2001)) that allow one to retrieve the hyperplanes in sublinear time, should be required.

The advantage of the method based on a classifier is the ability to identify the new significant associations that havent been derived from explicitly given co-occurrences of the features. This explicit analysis of the features is the approach used in the typical methods that are based on analyzing a subset of features in common (metrics A and B). Application of the classifier is much more flexible, based on it we can introduce different, significant, relations (metrics C and D) and if we select a proper metric we may achieve results very similar to the standard approaches (as it was in case of metrics A and E).

6. Results applied: semantic network

One very tempting feature of Wikipedia as an Information Technology project is its scale. Even after the aggressive filtering described in Section 4.1, the dataset used in experiments contains 145,000 categories. Each of those categories essentially represents some concept – abstract, like *dog*, or concrete, for example *Barack Obama*. The dataset, therefore, is a set of concepts larger than for example WordNet dictionary (Fellbaum, 1998), which contains about 117,000 synsets. Additionally, Wikipedia content constantly grows, as it is updated daily by editors all over the world.

Filtered Wikipedia categories with computed values of similarity can be assembled into a semantic network and visualized in this way. Moreover, with the use of the model computed for the metric (E) it is possible to enrich the network with keywords describing categories. Firstly, such a semantic network is a tangible outcome of our research presented in this paper. Secondly, visualization makes it easier for humans to browse and assess the results.

6.1. The metrics combined

The two highest quality metrics (A) and (E) were put together into one metric. So, both direct and aggregated versions of these

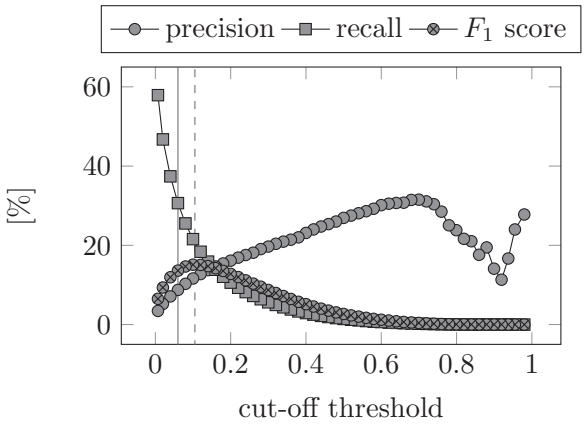

Fig. 16. Precision, recall and F_1 score depending on threshold. Dashed line marks peak F_1, continuous one the assumed cut-off threshold.

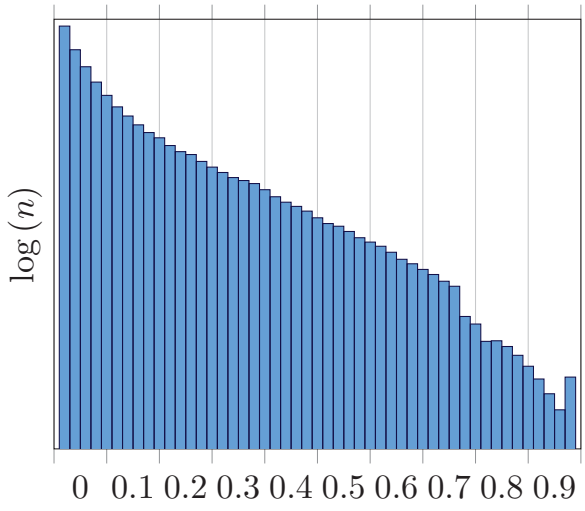

Fig. 17. Distribution of similarity weights in the combined metric. Axis y is scaled logarithmically.

metrics were used. In order to compute similarity weights for the output metric, the similarity value distributions were made equal. Then, the final similarities were computed as average values over individual metrics.

In order to keep the semantic network relatively small, it was necessary to set a proper cut-off threshold, i.e. boundary value of similarity, where categories in pairs are not related any more. To this end, once again the original Wikipedia relations were used as a reference set. Precision, recall and F_1 score were computed depending on the cut-off threshold. Fig. 16 presents obtained relationships between them.

Finally, the cut-off threshold was set to 0.06, since at this point the quality estimator F_1 starts declining fast.

A semantic network of 142,000 concepts and 1,133,000 relationships was finally obtained. Only 10% of those relations were original Wikipedia hierarchic connections. Such a low percentage illustrates the amount of relevant connections that are not marked in Wikipedia.

Fig. 17 presents distribution of similarity weights across all edges. Note that the number of related pairs of categories grow exponentially as the similarity value decreases.

6.2. Visualization

Lists of nodes and edges forming the semantic network were saved in two comma separated values files. Open Source graph visualization software Gephi (Bastian, Heymann, & Jacomy, 2009) was used to display and browse the network. Because of the large numbers of nodes and edges, *Ego Network* filter was applied, which makes only a selected category and its neighbors rendered. Fig. 18 presents sample visualizations obtained for category *Gdańsk*.

The graph used for visualization, presented in Fig. 18 has been based on aggregation of the proposed metrics. For now ,the visualisation is static in the sense that the only interaction with the user allows one to select the node (article) for presentation of relations. Definitively, it is worth to extend the proposed method with the functionality that allows us to select the particular metrics that define the type of the information the user is interested in. The data narrowed with selection of the particular similarity measure, should provide more precise information, and thus be more valuable for the user. For now, we have only weights that describe the strength of the relation, introduction of the semantically meaningful relations will strongly increase the readability of the data, especially if it very large scale. To do that, interpretation of the semantics of the relations created using particular metrics should be researched. The method for data presentation based on selecting particular types of relations should be very useful for increasing the efficiency of information retrieval in large scale repositories. Especially while it is dedicated for recommending the content that is related to the user preferences (Szymański & Duch, 2010).

7. Summary and further directions

Five similarity metrics (A)–(E) (defined in Section 3) were implemented and evaluated. Ideas underlying two of them – based on the assignment of articles to categories (A) and similarity between articles (B) – had been previously researched (Holloway et al., 2007; Pang & Biuk-Aghai, 2010) and (Szymański et al., 2013). Three others, which are formulated around the concept of using a multilabel text classifier, were original ideas that have not been published so far. The objective of our work was to compare all of these similarity measures.

Pang and Biuk-Aghai (2010) actually presents two methods. The first is used to compute direct similarities between categories. The second, making use of direct similarity and hierarchical relationships defined in Wikipedia, calculates aggregated similarities which are later recognized as final results of the metric. However, when proposing the algorithm the authors neither measure nor explain how the aggregation step influences the results obtained computing direct similarity. Thus, in order to conduct the experiments in a fair way, every metric was computed in two versions: direct and aggregated. Finally, comparison involving ten competing approaches was performed.

Comparison was based on original Wikipedia relationships coverage, mutual metric overlap, manual evaluation and workload required to compute a metric. According to the results, approach (A) was superior to the other metrics, followed by (E) which has similar quality but is very costly to compute, since it involves training a classifier using articles expressed as feature vectors. In the evaluation, both metrics (A) and (E) turned out to be very reliable, meaning that they could be successfully employed in a category links recommendation system – which would be particularly useful in cases when new categories are created.

Metric overlap test revealed that the approaches (A) and (E), different as they are, generate very similar results. Further analysis shows that in spite of being defined in a completely different way, metric (E) behaves similarly to (A). There is, however, a theoretical advantage to (E): it should be able to capture links between

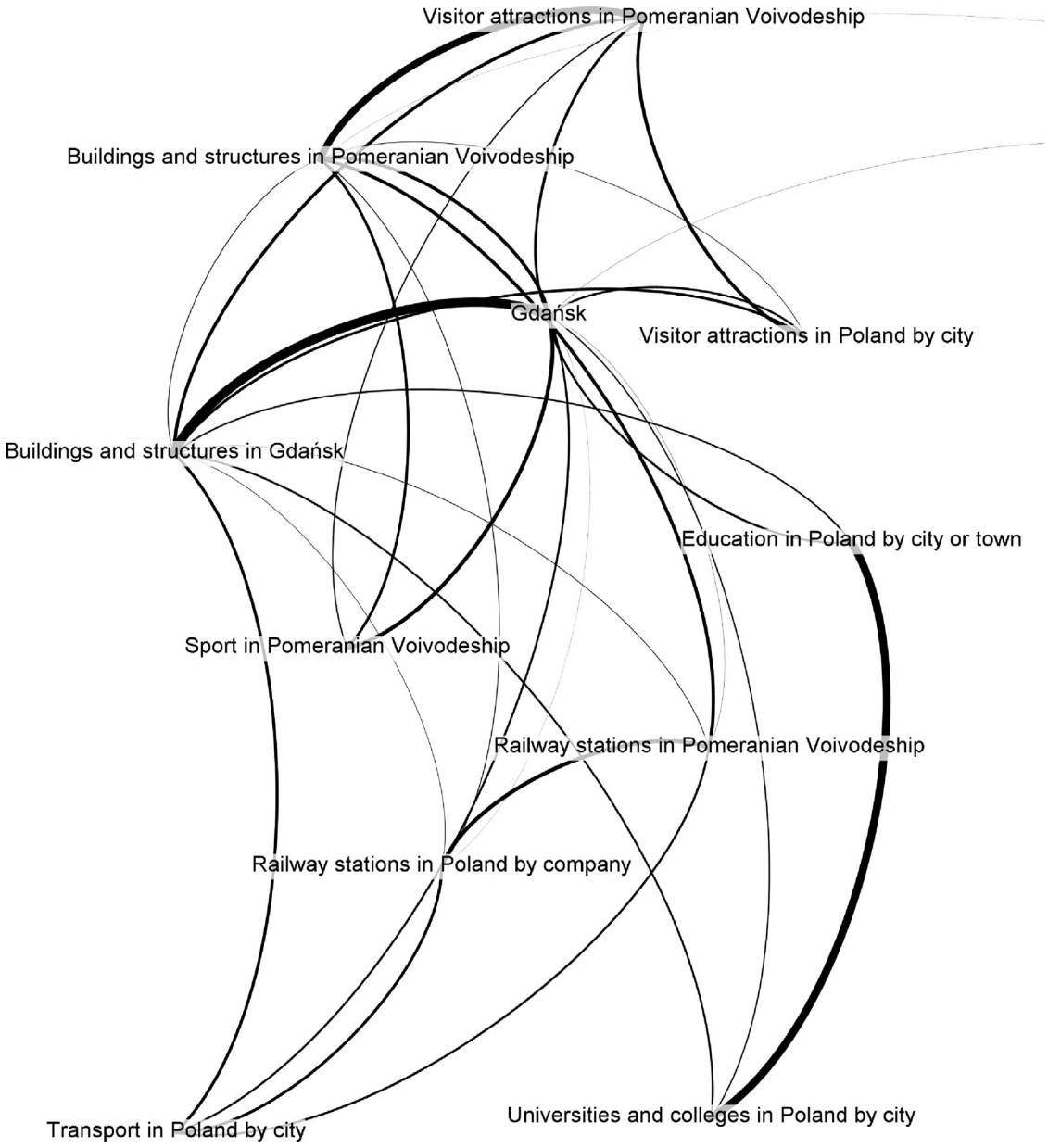

Fig. 18. Visualisation of categories similar to *Gdańsk*

categories when these contain similar articles, while (A) requires the same articles in both categories to state similarity. As it turned out, this advantage, while coming at a significant cost, did not provide much profit after all.

On the other hand, approach (E) might still be an avenue worth following. Inspection of the features used by the metric to define similarity between classes showed an apparent noise. Perhaps improving the stemming algorithm used to turn articles into stemmed words would increase the classifier's accuracy, which would result in a better similarity measure. Comparing its performance with implementations of the state-of-the-art Porter stemmer (Porter, 1997) is definitely worth a try.

One approach to computing similarities between categories that has not been touched upon in our research is to make use of links between articles. Results obtained in experiments presented in Szymański et al. (2013) were of low quality due to the very big number of links in Wikipedia: a significant fraction of links outgoing from a given article points to vaguely related pages. A promising alteration of the links-based strategy would be to consider links from infoboxes or *see also* sections, as these are generally more relevant than the average. It may, however, turn out that in such a scenario numbers of features per article are too low. In such a case, an approach combining word and link-based features (Aljaber, Stokes, Bailey, & Pei, 2010) might come in handy. Also, experiments with other representation methods especially *Explicit Semantic Analysis*(Gabrilovich & Markovitch, 2007) are definitely worth researching.

In the paper, the approach to relations mining has been based on employing the SVM classifier. Usage of other approaches for classification may also provide interesting results, and this is a natural extension of the proposed approach. One of the other interesting directions for continuing the research is employing the methods used typically in the recommendation systems. As an example we may consider the *Collaborative Filtering* (Breese, Heckerman, & Kadie, 1998) method that can be employed for analysis of the relations. In the typical application of *Collaborative Filtering*

recommendations are formed based on users' interests in particular products and their features. We can employ this approach, and in an analogical way, based on article features and their existing associations to the categories, a collaborative filtering for a given category will create a ranking of recommended ones. Considering the domain of information filtering systems, the proposed method of associations mining, can be also used for increasing the performance of product recommendations (Dabrowski & Acton, 2013). To do that, we need to have information about product classes and users. If such information is missing it can be introduced into the data eg.; by application of clustering algorithms.

The method proposed by us also has potential for extending the methods for missing data imputation. Based on identification of the relations between groups of similar data we can mine information that is missed. This can be particularly useful for increasing the density of connections in the graphs (Draszawka, Szymanski, & Krawczyk, 2014). Some algorithms, to operate require the strongly connected graphs, eg.: the *Page Rank* while it is computed using *Power Method*, needs to adapt the graph by adding some small values to the weights. The rationale, that we introduce new weights based on the analysis of the rations may lead to better results than adding some fixed values, that is the case of approach used in *Power Method*.

Acknowledgments

The work has been supported partially by the Polish Ministry of Science and Higher Education.

References

Adomavicius, G., & Tuzhilin, A. (2001). Using data mining methods to build customer profiles. *Computer, 34*(2), 74–82.

Agrawal, R., Imieliński, T., & Swami, A. (1993). Mining association rules between sets of items in large databases. *SIGMOD Rec., 22*(2), 207–216.

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th international conference on very large data bases*. In *VLDB '94* (pp. 487–499). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Aljaber, B., Stokes, N., Bailey, J., & Pei, J. (2010). Document clustering of scientific texts using citation contexts. *Information Retrieval, 13*(2), 101–131.

Bastian, M., Heymann, S., & Jacomy, M. (2009). Gephi: An open source software for exploring and manipulating networks. In *International aaai conference on weblogs and social media* (pp. 361–362).

Breese, J. S., Heckerman, D., & Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the fourteenth conference on uncertainty in artificial intelligence* (pp. 43–52). Morgan Kaufmann Publishers Inc.

Brin, S., Motwani, R., Ullman, J. D., & Tsur, S. (1997). Dynamic itemset counting and implication rules for market basket data. *SIGMOD Rec., 26*(2), 255–264.

Chen, S., Ma, B., & Zhang, K. (2009). On the similarity metric and the distance metric. *Theoretical Computer Science, 410*(24-25), 2365–2376.

Chernov, S., Iofciu, T., Nejdl, W., & Zhou, X. (2006). Extracting semantics relationships between wikipedia categories. *SemWiki, 206*.

Colgrove, C., Neidert, J., & Chakoumakos, R. (2011). Using network structure to learn category classification in wikipedia. 2014-01-09]. http://snap.stanford,edu/class/cs224w-2011/proj,.

Crammer, K., & Singer, Y. (2002). On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research, 2*, 265–292.

Dabrowski, M., & Acton, T. (2013). The performance of recommender systems in online shopping: A user-centric study. *Expert Systems with Applications, 40*(14), 5551–5562.

Draszawka, K., & Szymański, J. (2013). Thresholding strategies for large scale multi-label text classifier. In *Proceedings of the 6th international conference on human system interaction* (pp. 347–352). IEEE.

Draszawka, K., Szymanski, J., & Krawczyk, H. (2014). Towards increasing density of relations in category graphs. In *Intelligent tools for building a scientific information platform: From research to implementation* (pp. 51–60). Springer.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., & Lin, C.-J. (2008). Liblinear: A library for large linear classification. *Journal of Machine Learning Research, 9*.

Farina, D. L. J. (2010). Automatically assigning wikipedia articles to macrocategories. http://airwiki.ws.dei.polimi.it/index.php/WikipediaCategoryGraph.

Fellbaum, C. (Ed.). (1998). *WordNet: An electronic lexical database*. MIT Press.

Gabrilovich, E., & Markovitch, S. (2007). Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of the 20th international joint conference on artificial intelligence* (pp. 1606–1611).

Gantner, Z., & Schmidt-Thieme, L. (2009). Automatic content-based categorization of wikipedia articles. In *Proceedings of the 2009 workshop on the people's web meets nlp: Collaboratively constructed semantic resources* (pp. 32–37). Association for Computational Linguistics.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning* (1st ed.). Addison-Wesley Professional.

Hearst, M. A., Dumais, S., Osman, E., Platt, J., & Scholkopf, B. (1998). Support vector machines. *Intelligent Systems and their Applications, IEEE, 13*(4), 18–28.

Hoffart, J., Suchanek, F., Berberich, K., Kelham, E., de Melo, G., Weikum, G., … Pease, A. (2009). Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Communications of the ACM, 52*(4), 56–64.

Holloway, T., Bozicevic, M., & Börner, K. (2007). Analyzing and visualizing the semantic coverage of wikipedia and its authors: Research articles. *Complex., 12*(3), 30–40.

Hsu, C.-W., & Lin, C.-J. (2002). A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on, 13*(2), 415–425.

Joachims, T. (1998). *Text categorization with support vector machines: Learning with many relevant features*. Springer.

Kai Bo Duan, & Sathiya Keerthi, S. (2005). Which is the best multiclass svm method? an empirical study. In *Proceedings of the sixth international workshop on multiple classifier systems* (pp. 278–285).

Kazienko, P. (2009). Mining indirect association rules for web recommendation. *International Journal of Applied Mathematics and Computer Science, 19*(1), 165–186.

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., … Bizer, C. (2013). Dbpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*. Under review.

Lin, W., Ruiz, C., & Alvarez, S. A. (2000). A new adaptive-support algorithm for association rule mining. *Technical Report*. Department of Computer Science of Worcester Polytechnic Institute.

Luis, V. A., Kedia, M., & Blum, M. (2006). Verbosity: a game for collecting common-sense facts. In *Proceedings of the sigchi conference on human factors in computing systems*. In *CHI '06* (pp. 75–78). New York, NY, USA: ACM.

Moro, A., & Navigli, R. (2012). Wisenet: building a wikipedia-based semantic network with ontologized relations. In *Proceedings of the 21st acm international conference on information and knowledge management*. In *CIKM '12* (pp. 1672–1676). New York, NY, USA: ACM.

Nastase, V., & Strube, M. (2013). Transforming wikipedia into a large scale multilingual concept network. *Artificial Intelligence, 194*, 62–85.

Pang, C.-I., & Biuk-Aghai, R. P. (2010). A method for category similarity calculation in wikis. In *Proceedings of the 6th international symposium on wikis and open collaboration* (pp. 191–192). ACM.

Perez, B., Feo, C., West, A. G., & Lee, I. (2012). A graph-based algorithm for categorizing Wikipedia articles. *Technical Report*. University of Pennsylvania.

Porter, M. F. (1997). An algorithm for suffix stripping. In K. Sparck Jones, & P. Willett (Eds.), *Readings in information retrieval* (pp. 313–316). Morgan Kaufmann Publishers Inc.

Rifkin, R., & Klautau, A. (2004). In defense of one-vs-all classification. *Journal of Machine Learning Research, 5*, 101–141.

Salton, G., & McGill, M. J. (1986). *Introduction to modern information retrieval*. New York, NY, USA: McGraw-Hill, Inc.

Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM, 18*(11), 613–620.

Schonhofen, P. (2006). Identifying document topics using the wikipedia category network. In *Proceedings of the 2006 ieee/wic/acm international conference on web intelligence*. In *WI '06* (pp. 456–462). Washington, DC, USA: IEEE Computer Society.

Singh, P., Lin, T., Mueller, E. T., Lim, G., Perkins, T., & Zhu, W. L. (2002). Open mind common sense: Knowledge acquisition from the general public. In *On the move to meaningful internet systems 2002: Coopis, doa, and odbase* (pp. 1223–1237). Springer-Verlag.

Speer, R., & Havasi, C. (2012). Representing general relational knowledge in conceptnet 5. In N. C. C. Chair), K. Choukri, T. Declerck, U. Doğan, B. Maegaard, J. Mariani, J. Odijk, & S. Piperidis (Eds.), *Proceedings of the eighth international conference on language resources and evaluation (lrec'12)* (pp. 3679–3686). Istanbul, Turkey: European Language Resources Association (ELRA).

Strube, M., & Ponzetto, S. P. (2006). Wikirelate! computing semantic relatedness using wikipedia. In *Proceedings of the 21st national conference on artificial intelligence - volume 2* (pp. 1419–1424). AAAI Press.

Szymański, J. (2010). Mining relations between wikipedia categories. In *Proceedings of the second international conference on networked digital technologies (part ii)* (pp. 248–255). Springer.

Szymański, J. (2010). Towards automatic classification of wikipedia content. In *Proceedings of the 11th international conference on intelligent data engineering and automated learning*. In *IDEAL'10* (pp. 102–109). Berlin, Heidelberg: Springer-Verlag.

Szymański, J. (2013). Wikipedia Articles Representation with Matrix'u. In *Distributed computing and internet technology*. In *Lecture Notes in Computer Science: 7753* (pp. 500–510). Springer Berlin Heidelberg.

Szymański, J. (2014). Comparative analysis of text representation methods using classification. *Cybernetics and Systems, 45*(2), 180–199.

Szymański, J., Deptuła, M., & Krawczyk, H. (2013). Identyfikacja powiazań pomiedzy kategoriami wikipedii z użyciem miar podobieństwa artykułów. *Studia Informatica, 34*(2A).

Szymański, J., & Duch, W. (2010). Dynamic semantic visual information management. In *Proceedings of the 9th international conference on information and management sciences* (pp. 107–117).

Von Ahn, L., Maurer, B., McMillen, C., Abraham, D., & Blum, M. (2008). re-CAPTCHA: Human-Based Character Recognition via Web Security Measures. *Science, 321*(5895), 1465–1468.

Wired 14.06 (2013). The rise of crowdsourcing. http://www.wired.com/wired/archive/14.06/crowds.html. Accessed: 2014-05-19.

Yu, C., Ooi, B. C., Tan, K.-L., & Jagadish, H. (2001). Indexing the distance: An efficient method to knn processing. In *Vldb: vol.1* (pp. 421–430).