Improving Folksonomies Quality by Syntactic Tag Variations Grouping

Francisco Echarte, José Javier Astrain, Alberto Córdoba, Jesús Villadangos
Dpt. Ingeniería Matemática e Informática
Campus de Arrosadía, Pamplona (Spain)
patxi@eslomas.com, {josej.astrain, alberto.cordoba, jesusv}@unavarra.es

ABSTRACT

Folksonomies offer an easy method to organize information in the current Web. This fact and their collaborative features have derived in an extensive involvement in many Social Web projects. However they present important drawbacks regarding their limited exploring and searching capabilities, in contrast with other methods as taxonomies, thesauruses and ontologies. One of these drawbacks is an effect of its flexibility for tagging, producing frequently multiple variations of a same tag. In this paper we propose a method to group syntactic variations of tags using pattern matching techniques. We propose the utilization of a fuzzy similarity measure and we conclude that this technique offers better results than other classic techniques after comparing them on a large real dataset.

Categories and Subject Descriptors

H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval – *information filtering.*

General Terms

Measurement, Performance, Design, Experimentation.

Keywords

Folksonomies, Semantic Web, Fuzzy Logic, Pattern Matching.

1. INTRODUCTION

Folksonomies [10] offer users an easy way to sort and organize information by assigning text tags at different resources, such as photos, web pages, documents, etc. User annotations and categorization define collaboratively the semantics of the tags and resources used. One of the key success factors of folksonomies is its simplicity of use. Nevertheless, this ease of use has also some important disadvantages. Its main drawback is related to the ability of users to assign tags freely. This produces the absence of any a priori structure among tags, and allows users to use synonyms, syntactic tag variations, different granularity levels [2], lowering the quality of folksonomies and making more difficult the exploration and finding of information [4,8]. Reducing syntactic variations aids to improve the quality of folksonomies.

SAC'09, March 8-12, 2009, Honolulu, Hawaii, U.S.A.

Syntactic tag variations can be caused by typographical mistakes (*semanticweb*, *semnticwev*, *zemantcweb*); or by using the singular or plural of the same word (*semanticweb*, *semanticwebs*); or by using separators (*semantic-web*, *semanticweb*); or a combination of them (semntic-web, *smanticweb*, *semntic-webs*, etc.). Although syntactic tag variations may have the same sense, resources would be classified under different labels. This fact makes more confusing the searching and browsing processes, difficulting the location of information and the navigation on the folksonomy. However, identifying all of them as variations of the same label "*semantic web*" and grouping them under the same tag, a user could access this tag obtaining all the information concerning the resources associated with it and its syntactic variations. Tag Clouds for example could show a representative of each group of related variations and provide access to all resources annotated with them. This also could be used for searching and browsing.

This paper focuses on improving folksonomy quality by the application of a method based on pattern matching techniques in order to group syntactic variations of tags under a unique tag. The method extends the folksonomy model including tag variations. In addition, the method uses a selection step to discriminate between *new* or *known* tags. The method is flexible enough to apply different techniques at the discriminator. In this way, we apply and compare some of the most relevant pattern matching techniques Generalized Levenshtein Distance [7], Hamming Distance [5] and a fuzzy similarity measure based on fuzzy automata with transition by empty string [1]. We propose a way to introduce variable costs for edit operations used to transform an observed string in a pattern string and conclude that fuzzy similarity measures provide better results than the rest of the analyzed techniques. Conclusions are obtained by testing the method for a large data subset extracted from a known folksonomy, CiteULike (http://www.citeulike.org).

There are several works in the Literature focused on solving some of the problems associated with folksonomies. Most of these proposals do not take into account that a relevant number of the existing tags corresponds to syntactic variations (erroneous or not) of previously existing tags. The performance of a pre-filtering of the tags before applying an algorithm for tag clustering, as occurs in [9], allows minimizing the effects of syntactic variations increasing the quality of tag clustering. In [9] Specia and Motta create clusters of semantically related tags over a reduced experimental data set, using a previous step in which Levenshtein similarity measure is used to reduce the number of tags identifying syntactic variations, replacing each

identified variation by a representative tag. Another way to represent these variations is presented in [2] where an ontology with three properties associated to tags (*prefLabel*, *altLabel* and *hiddenLabel*) is used.

In [1] we propose a fuzzy automaton with ε-moves ($FA\varepsilon$) and constant costs for edit operations. It includes other classical measures and takes into account the context (fuzzy states of the automaton) of the pattern and candidate strings. This context allows dealing with both edition errors (insertion, deletion and change) and transposition errors (typo errors). Such automaton can be applied to identify syntactic variations of tags. In this paper we introduce a modification to consider variable costs in the automaton. From our knowledge, it is the first time that fuzzy automata are proposed to group syntactic tag variations on folksonomies.

The rest of the paper is organized as follows: section 2 presents a folksonomy definition adapted to represent syntactic variations of tags; section 3 shows a method to group tags and their syntactic variations and introduces an imperfect pattern matching technique based on fuzzy-logic; section 4 describes the experimental scenario; section 5 analyzes the experimental results; finally, conclusions and references end the paper.

2. Syntactic Variations Grouping

A folksonomy F can be defined as: $F=(U,R,T,f_a:UxRxTx\ldots xT)$, being U, R and T the finite sets for the users, resources and tags defined in the folksonomy, respectively; and being f_a the annotation relation, which relates a user, with a resource and with a set of tags used by the user to annotate the resource.

We propose an extension of this definition in order to represent the syntactic variations of tags. A folksonomy is defined as the tuple: $F'=(U,R,T,T',f_a:UxRxTx\ldots xT,f_g:T'\rightarrow Tx\ldots xT)$. In this model U, R and T keep their meaning and a new set with name T' is used to represent the grouping of T elements, being $T'\subseteq T$. This model allows representing tag variations as members of set T and grouping them under T' elements. On one hand, relation f_a keeps the same meaning, relating a user with a resource and a set of tag variations used to annotate the resource by the user. On the other hand, function f_g represents the relation between T' groups of tags and T tags variations. For example, the set T could contain the following subset of tags {*semantic-web*, *semantics-web*, *semtic-wb*}. Then, one element of T' could be *semantic-web*, representing these tags, and f_g could represent the relation between the T' element and its tags variations.

3. Tag grouping method

Through annotation processes, folksonomies are updated with user annotated resources and tags. When the user annotates a resource, then a set of elements, if necessary, are added to the folksonomy: the user, the resource, the annotation and the tags. However, annotation processes are not worried about syntactic tag variations, and non-duplicated tags are automatically added.

We propose a method that extends this behaviour adding a new component: a tag discriminator. This component determines if a tag used in an annotation is new, or if it is a syntactic variation of an already known tag in the folksonomy. It uses a dictionary built upon the different tag groups (T' elements) identified.

In our method, as input we have the set of new tags used in an annotation and as output, the proposed folksonomy model is updated with this information, creating new groups of tags, or grouping new tag variations under already known tag groups. The tag discriminator component is used to detect whether the tags are new, or syntactic variations of other tags.

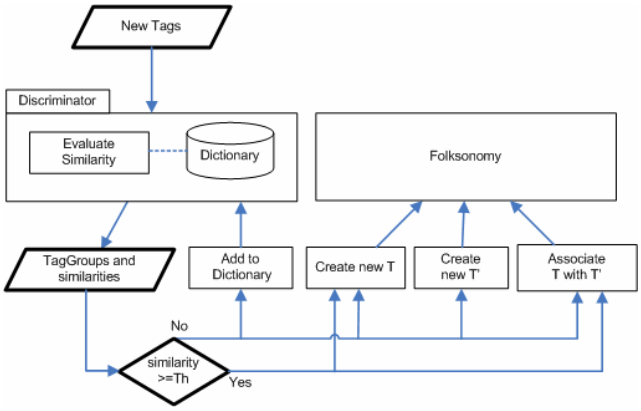

Figure 1. Flow diagram of the proposed Method.

Figure 1 shows the steps of the method. The new detected tags are provided to the tag discriminator. The discriminator evaluates the similarity of each tag with the tag group elements defined in the dictionary. As result the discriminator returns the nearest tag group and the similarity value for each new tag. Based on this information a selection criteria based on a threshold (Th) is used. Whenever the similarity is greater or equal than the threshold, the new tag is considered as a variant of a known tag; in other case, it is considered as a new tag. When the tag is identified as a new variant of a previously defined T' element, a new T element is created in folksonomy and it is associated with it. When the tag is identified as new, then new T' and T elements are created and associated.

The described method allows the use of any pattern matching technique for the identification of syntactic variations of tags. In this paper, we propose the utilization of a fuzzy similarity measure with variables costs and we compare it with other classic measures verifying that our proposal offers better results.

Approximate string matching allows considering typing, spelling and swapping errors, closely related vocabularies, word mutations, and many other situations which are quite common in knowledge tagging. Distance between an observed string and the pattern is therefore computed as the minimum distance between the candidate and a fixed string matching the regular expression. The classical Hamming distance computes the distance between two strings of equal length, measuring the minimum number of *errors* that transformed α in ω. This hard restriction makes difficult to deal with tags of different lengths. In the same way, when dealing with imprecise information and/or with unknown error model, the use of fuzzy logic can improve the detection achieved. A fuzzy automaton with transitions by empty string FA_ε [1] can provide the classification of strings containing any amount of edit errors, where the error model does not exists or it is unknown. Fuzzy finite state automaton with ε-moves, denoted FA_ε, allow fuzzy automaton movements by empty string imply a fuzzy state change without consuming any input symbol.

Edit operations allow transforming the observed string into the pattern string. Each edit operation has its own associated cost. Usually, these costs are fixed to a certain value for the whole universe of symbols considered. However, in this paper we propose to make these costs a function of some parameters as the symbols involved in the operation, the length of the strings, the position of the symbols in the string and so on. The costs are selected with the goal to minimize the effects of singular and plural, typographical and transposition misspellings and separators. Variable costs have been defined after analyzing the measures on the experimental datasets, adapting experimentally the fuzzy measure to deal better with these variations.

A fuzzy finite state automaton with ε-moves FA_ε, is a sixtuple $(Q,\Sigma,\mu,\mu_\varepsilon,\sigma,\eta)$ where Q is a non-empty finite set of states; Σ is a non-empty finite set of input symbols (*input alphabet*) where Σ^+ is the set of all non-empty strings over Σ, and $\Sigma^* = \Sigma^+ \cup \{\varepsilon\}$; μ: $Q \times Q \times \Sigma \rightarrow [0,1]$ is the state transition function; σ and η are fuzzy sets on Q; and μ_ε is a reflexive binary fuzzy relation on Q representing the state transition function by empty string. For q, $p \in Q$ and $x \in \Sigma$, the value $\mu(q,p,x) \in [0,1]$ represents the degree to which the automaton in state q and with the input symbol x may enter to state p. For $q \in Q$, σ (q) indicates the degree to which q is an initial state, and $\eta(q)$ indicates the degree to which q is a final state.

(1) $\hat{\mu} : \Im(Q) x \Sigma \rightarrow \Im(Q)$

(2) $\hat{\mu}_\varepsilon : \Im(Q) \rightarrow \Im(Q)$ is the fuzzy state transition function by empty string. Given a fuzzy state $V \in \Im(Q), \hat{\mu}_\varepsilon(V) = V \circ \mu_\varepsilon^T$, where μ_ε^T is the T-transitive closure of μ_ε.

(3) $\mu^* : \Im(Q) x \Sigma^* \rightarrow \Im(Q)$ is the extended transition function for the fuzzy finite state automaton with ε-moves. It is defined as follows:

a) $\mu^*(V,\varepsilon) = \hat{\mu}_\varepsilon(V) = V \circ \mu_\varepsilon^T, \forall V \in \Im(Q)$

b) $\mu^*(V,\alpha x) = \hat{\mu}_\varepsilon(\hat{\mu}(\mu^*(V,\alpha),x)) = (\mu^*(V,\alpha) \circ \mu[x]) \circ \mu_\varepsilon^T,$
$\forall V \in \Im(Q), \alpha \in \Sigma^*, \text{and } x \in \Sigma$

The language accepted by FA_ε, denoted $L(FA_\varepsilon)$, is the fuzzy set on Σ^* such that

$$L(FA_\varepsilon)(\alpha) = \max_{q \in Q}\left(\mu^*(\sigma,\alpha)(q) \circ \eta(q)\right), \forall \alpha \in \Sigma$$

The variable costs used in the fuzzy measure derive from the fixed costs (0.005, 0.001 and 0.01 for the change, deletion and insertion respectively) described in [3]. In order to deal with the variations described above, we have experimentally selected a set of costs for each kind of variation[1]. Insertion, deletion and change functions depend on three characteristics: (1) the characters of the pattern and observed strings to check at each step of the measure; (2) the two characters around them at left and right; and (3) the lengths of each string. Thus cost functions can detect the transposition of adjacent characters, changes in the termination of strings about plurals and singulars, and the substitution or elimination of separators.

4. Workbench

This section describes the experimental scenario we have used to evaluate our proposal, paying special attention to the datasets and the methodology. This workbench is available on the web[1] .

[1] http://www.eslomas.com/index.php/publicaciones/tagsfuzzy

We have collected data from the social web CiteULike in order to evaluate our proposal, collecting a total number of 2,290,740 annotations. After a first analysis of the data set, two tags with a significantly larger number of annotations than the rest, probably generated by any automatic procedure, have been deleted: "*bibtex-import*" and "*no-tag*". The resulting data set has the following characteristics: (1) 2,038,172 annotations, (2) 494,206 resources, (3) 21,480 users, and (4) 151,522 tags.

In order to evaluate our proposal we have created two data sets: one with the aim of checking the correct identification of variations (DS1) and another (DS2) to validate the proposed method focusing the attention on the discriminator operation to recognize new tags.

DS1 is obtained from the 10,000 most often used tags. These tags are used in 1,557,198 annotations, representing the 76.4 % of the total amount of annotations. DS1 consists of a set of tuples <*pattern tag-candidate tag*>: *pattern* is one of the 10,000 related tags, and *candidate* is a syntactic variation of the *pattern*. These variations are created automatically considering different cases: (i) the singular or plural, (ii) simulation of a typographical error, (iii) simulation of a transposition of symbols, (iv) removal and replacement of separators and (v) the own pattern tag in order to verify that similarity measures do not introduce new errors. In the creation process, if a syntactic variation of a pattern tag *t* fits another pattern tag *t'*, the candidate tags obtained from *t'* are addressed to *t* and *t'* is deleted. After the whole process, DS1 contains 8,806 different pattern tags and 39,255 tuples *(pattern, candidate)* to check.

DS2 contains 5,000 tags not included as pattern tags in DS1. These tags are used in 122,394 annotations representing the 6% of the total amount of annotations. We create a dictionary with the 8,806 pattern tags contained in DS1. This dictionary is used to perform the Levenshtein, Hamming and fuzzy measures of similarity (distance) over DS1 and DS2 datasets.

5. Methodology

In our proposal, the grouping of syntactic variations of tags becomes useful whenever: (1) the pattern matching techniques applied at the discriminator ensure a high recognition rate of tags, which are variations of an existing one; and (2) identify, with a high degree of success, new tags that do not fit any existing one. The goal is to maximize the number of syntactic tag variations identified without conditioning the recognition of new tags. In our experimental scenario, the goal is to maximize the number of correctly identified tuples <*pattern, candidate*> on DS1; and to maximize the number of tags identified as new tags on DS2. In order to prove this behaviour, we apply the fuzzy automaton with transitions by empty string (FA_ε), the GLD distance and the Hamming distance in our experiments.

Note that the discriminator provides the pattern tag of the dictionary with the greater similarity measure to the input tag. We denote in the following by *candidate* this input tag and by *pattern'* the discriminator output as depicted in Figure 2.

We take the *candidate* tag from each tuple in DS1 and we apply the above algorithms over the dictionary at the discriminator. Note that the algorithm could select a pattern tag (*pattern'*) different to the correct pattern (the associated to the *candidate* tag in DS1). In addition, the discriminator provides the similarity (distance) values between *candidate* and *pattern'*. In

order to interpret the results of the experiment, we denote by *OK* the case when the tag selected by the algorithm (*pattern'*) is the *pattern* associated to *candidate* tag in DS1. That is, the algorithm selects the expected pattern tag. We use *NOK* to describe the case when *pattern'* and *pattern* do not fit.

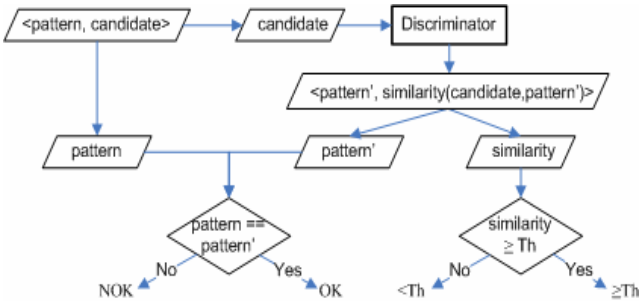

Figure 2. Methodology schema.

A threshold level (*Th*) determines the accuracy of the discriminator. The candidate tag is classified as a syntactic variation of the *pattern'* tag if the discriminator provides a similarity value higher or equal than this threshold. Thus, for example *NOK & ≥ TH* indicates that *candidate* and *pattern'* tags match with a high degree of similarity, but *candidate* derives from a *pattern* tag (DS1) which is different to *pattern'*.

Dealing with the problem of new tags identification, when the similarity values obtained for the tags contained in DS2 (*candidates*) are lower than *Th*, tags are considered as new pattern tags (*New*). In other case tags will be considered as syntactic variations (*Not New*) of the *pattern'* provided by the discriminator.

6. Experimental Results

Table 1 shows the results of processing data sets DS1 and DS2 with the three measures, using a dictionary with the 8,806 distinct tags of DS1. Results on DS1 are shown in 4 different columns, representing if the correct pattern has been identified for each candidate string (*OK*) or not (*NOK*), and if the similarity value is greater than a determined threshold (*≥Th*).

Table 1. Results on data sets DS1 and DS2.

	DS1				DS2	
	NOK & < Th	**NOK & ≥ Th**	**OK & <Th**	**OK & ≥Th**	**< TH *New***	**≥TH *Not New***
Hamming	3,095	2,152	8,127	25,881	4,162	838
Levenshtein	1,641	2,516	7,767	27,331	4,016	984
Fuzzy	753	2,608	8,655	27,239	4,016	984
Fuzzy VC	89	1,326	403	37,437	4,071	929

Threshold values for each measure have been obtained experimentally, being 1.0 for Levenshtein and Hamming measures, 0.001 for the fuzzy measure using constant costs for each edition operation, and 0.003 for the fuzzy meassure with variable costs (Fuzzy VC). Table 1 shows that Hamming measure has lightly worst results at the recognition pattern variations in DS1 (*OK & ≥Th*) and that it does not overcome the similarity or distance threshold in more cases than the two other measures. This shows that Hamming tends to identify less variations and therefore to identify them as new tags, what corresponds with the results on DS2, where it has a greater identification ratio than the other measures. Levenshtein and fuzzy measures have similar results on DS1 and identical on DS2. The four measures identify correctly the pattern and candidate strings when they coincide, used to prove that measures do not introduce new errors.

Comparing fuzzy measures, it can be seen that the ratio of success *OK & ≥Th* has increased significantly from 27,239 to 37,437. This is mainly due to the reduction of strings correctly identified but that were lower than the threshold value. This improvement has been achieved without adversely affecting the results on DS2, which have been even improved.

It also can be seen that the number of incorrect identifications between candidate and pattern strings (*NOK & ≥Th*) has been reduced almost to the half, from 2,608 to 1,326.

Table 2 shows a breakdown of the fuzzy with variable costs measure results on DS1 by variation type.

Table 2. Fuzzy VC results on DS1

	NOK & < Th	**NOK & ≥Th**	**OK & <Th**	**OK & ≥Th**
Self	0	0	0	8,788
Plural/Singular	17	170	207	8,581
TypoError	24	752	101	8,076
Transposition	48	400	88	8,233
Separators	0	4	7	3,759
Total (#)	89	1,326	403	37,437
Total (%)	0.23%	3.38%	1.02%	95.37%

The greater difference between Fuzzy and Fuzzy VC measures corresponds to transposition variations, where only 21 out of 8,769 were recognized correctly with Fuzzy measure, comparing with the 8,233 recognised by Fuzzy VC measure. The reason is that the fuzzy measure uses two changes of character to solve the transposition in the candidate string, so this produces that the similarity was lower than the threshold value with Fuzzy measure. However Fuzzy VC measure is able to deal with this situation. Regarding plural/singular variations Fuzzy measure identified correctly 7,022 variations out of 8,975 and Fuzzy VC improves this result to 8,581 correct identifications.

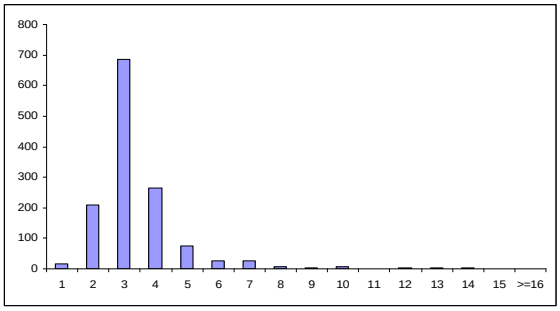

Figure 3. Fuzzy VC failures on DS1 per string length

Figure 3 shows a detail of these failures according to the string lengths. It shows that there are a predominant percentage of errors associated to string lengths lower or equal than four. The reason is that any variation in short length strings can produce

strings more similar to other patterns distinct of the original. For instance, comparing candidate string *lzb* and *wark*, created from *lab* and *work* patterns, the resultant strings have been *lsb* and *dark*, instead of the original patterns *lab* and *work*.

With the objective of ignoring strings with lengths lower or equal than 3 (they will always be considered as new tags and will not be grouped), information about the amount of tags in the initial dataset of CiteULike have been obtained. Based on these data, trying to identify variations only for lengths greater than 3, the measure would be used for a 95.37% of the tags in the initial CiteULike data set, corresponding to a 91.07% of the total number of annotations.

Table 3 summarizes the results obtained applying the 4 measures on DS1 and DS2 data sets, ignoring tags with length lower or equal than 3. It shows that the identification errors (*NOK & ≥Th*) of Fuzzy VC measure have been reduced from 1,326 to 324. Moreover it shows also that Fuzzy VC measure has a success percentage identifying new tags in the folksonomy on DS2 of 88.39% (4,056 out of 4,589). Table 4 shows that the correct identifications percentage (*OK & ≥Th*) is close to 98% (97.36% omitting "*Self*" variations").

Table 3. Results ignoring lengths ≤3.

	DS1				DS2	
	NOK & < Th	**NOK & ≥ Th**	**OK & <Th**	**OK & ≥Th**	**< TH (New)**	**≥TH (Not New)**
Hamming	2,966	587	8,111	24,431	4,171	418
Levenshtein	1,539	871	7,751	25,934	4,076	513
Fuzzy	646	874	8,644	25,931	4,050	539
Fuzzy (VC)	40	324	373	35,358	4,056	533

Table 4. Fuzzy VC results ignoring lengths ≤3 on DS1.

	NOK & < Th	**NOK & ≥Th**	**OK & <Th**	**OK & ≥Th**
Total (#)	40	324	373	35.328
Total (%)	0.11%	0.90%	1.03%	97.96%

Because of the DS2 data set has been created automatically from a subset of the CiteULike tags, without any a priori revision, it is possible that some of the tags correspond with variations of some other known tags in DS1. We have checked tags identified as *New* on DS2 with a similarity measure (Fuzzy VC) in the interval [0.0000001, 0.003] verifying that there effectively new tags. We have also checked manually the 533 results identified as variations on DS2 obtaining that 155 tags correspond to syntactic variations of patterns present in DS1, and so in the dictionary. This produces that the real success ratio increases to 91.76% (4,211 out of 4,589) and the incorrect ratio decreases to 8.24% (378 out of 4,589). Most of the 378 incorrect identifications have been also detected that corresponds with changes at one character, mainly at tags with small length. These kind of change produce that a valid tag is transformed in another valid tag, as *flag* and *flap*, *that* and *chat*, or *holography* and *homography*. The measure cannot do anything in this situation because it recognizes the changed tag as a valid pattern that exists in the dictionary.

To improve these results other techniques could be used: confusion matrices, greater costs to changes depending on the candidate string lengths, taking into account co-occurrences of tags, or the utilization of external services to validate the correctness of tags.

7. Conclusions

A pattern matching measure based in a fuzzy similarity measure has been tested on a large dataset in two different ways: (i) identifying pattern-candidate combinations and (ii) identifying new tags. Its behaviour has been also compared with two other classical methods, as Hamming and Levenshtein. The experiments show that the fuzzy measure with variable costs, gets significantly better results than these other measures, and that it can be applied to large datasets getting high success ratios, near 98% correctly identified variations of known tags and near 92% correctly identified new tags, in the experiments performed ignoring tags with lengths lower than four.

Based on this pattern matching techniques, a method to group syntactic tag variations transparently has been proposed. This method allows identifying tag variations and grouping them. We consider that grouping could improve navigation and searching capabilities in folksonomies and could also be used to improve other techniques, as clustering or automatic suggestions, focused in solve other folksonomies problems.

Acknowledgements

Supported by the Spanish Res.Council TIN2006-14738-C02-02.

8. References

[1] Astrain, J.J., González de Mendívil, J.R., Garitagoitia, J.R.: Fuzzy automata with ε-moves compute fuzzy measures between strings, *FSS*, 157, 11 (2006), 1550—1559.

[2] Echarte, F., Astrain, J.J., Córdoba, A., Villadangos, J.: Ontology of Folksonomy: A New Modeling Method. In SAAKM 2007, (Whistler, Canada, October 28-31, 2007).

[3] Garitagoitia, J.R., González de Mendívil, J.R., Echanobe, J., Astrain, J.J., Fariña, F.: Deformed Fuzzy Automata for Correcting Imperfect Strings of Fuzzy Symbols, IEEE Transactions on Fuzzy Systems, 11, 3 (2003), 299-310.

[4] Guy, M., Tonkin, E.: Folksonomies - Tidying up Tags? DLib Magazine, 12, 1 (2006).

[5] Hamming, R.W.: Error Detecting and Error Correcting Codes, Bell System Tech. Journal, 26, 2 (1950), 147-160.

[6] Hassan-Montero, Y., Herrero-Solana, V.: Improving tag-clouds as visual information retrieval interfaces. In Int. Conference on Multidisciplinary Information Sciences and Technologies (Mérida, Spain, Oct., 2006).

[7] Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals, Soviet Physics Doklady, 10, 8 (1966), 707–710.

[8] Mathes, A.: Folksonomies - Cooperative Classification and Communication Throught Shared Metadata. Computer Mediated Communication, Dec (2004).

[9] Specia, L., Motta, E.: Integrating Folksonomies with the Semantic Web. In European Semantic Web Conference, LNCS 4519, Springer. (Heidelberg, 2007), 503—517.

[10] Vander Wal, T.: Folksonomy, http://vanderwal.net/folksonomy.html