A Semantic Clustering-Based Approach for Searching and Browsing Tag Spaces

Damir Vandic
vandic@ese.eur.nl

Jan-Willem van Dam
jwvdam@gmail.com

Frederik Hogenboom
fhogenboom@ese.eur.nl

Flavius Frasincar
frasincar@ese.eur.nl

Econometric Institute
Erasmus University Rotterdam
PO Box 1738, NL-3000 DR
Rotterdam, the Netherlands

ABSTRACT

Many of the existing cloud tagging systems are unable to cope with the syntactic and semantic tag variations during user search and browse activities. As a solution to this problem, in this paper, we propose the Semantic Tag Clustering Search, a framework able to cope with these needs. The framework consists of three parts: removing syntactic variations, creating semantic clusters, and utilizing the obtained clusters to improve search and exploration of tag spaces. For removing syntactic variations, we use the normalized Levenshtein distance, and the cosine similarity measure based on tag co-occurrences. For creating semantic clusters, we improve an existing non-hierarchical clustering technique. Using our framework, we are able to find more clusters and achieve a higher precision than the original method. The advantages of a cluster-based approach for searching and browsing through tag spaces have been exploited in Xplore-Flickr.com, the implementation of our framework.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*clustering*

General Terms

Languages, design, management

Keywords

Tag spaces, semantic clustering, syntactic variations, Flickr

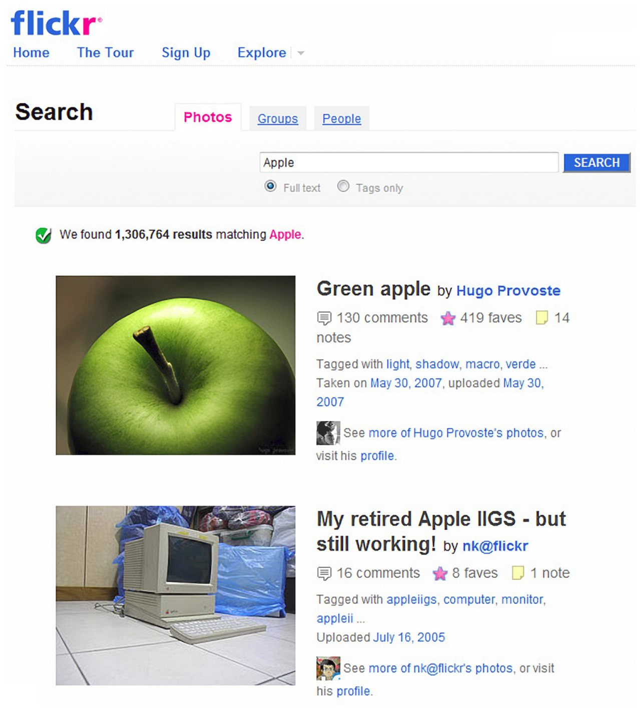
1. INTRODUCTION

Today's Web offers many services that enable users to label content on the Web by means of tags. Flickr and Delicious (also known as del.icio.us) are two well-known applications utilizing tags. In this paper, we focus on the Flickr Service, but our results can easily be applied to other social tagging systems. Registered Flickr users are allowed to upload and tag photographs. As with most tagging systems the user has no restrictions on the tags that can be used, i.e., the user can use any tag to his or her likings.

Even though tags are a flexible way of categorizing data, they have their limitations. Tags are prone to typographical errors or syntactic variations due to the amount of freedom users have. This results in different tags with the similar meanings, e.g., 'waterfal' and 'waterfall'. A query for 'waterfall' on Flickr returns $1,158,957$ results, whereas 'waterfal' returns $1,388$ results. This implies that potentially $1,157,569$ results are lost due to a typographical mistake. These syntactic variations of tags are very important aspects to consider when designing a search engine. Google, for example, has auto syntactic variation detection in their search service.

Users also describe pictures in different ways. For a picture which shows the interior of a house, most users would use the tag 'interior', where others would use a tag like 'inside' or 'furniture', these tags being semantically related. When someone searches for 'furniture', they are probably also interested in pictures tagged with 'interior'. Other common issues with tagging are related to the use of synonyms. These synonyms result in different results when searching, exploring, or retrieving information from a tagging system like Flickr. Homonyms can also occur, e.g., 'Apple' refers to pictures related to the brand as well as pictures related to an apple growing on a tree, as shown in Figure 1. The search engine cannot distinguish the multiple meanings the word 'Apple' can have.

For many applications, these symptoms are a problem. There is no structure, hierarchy, or classification available in most tagging systems. A lot of applications could benefit from the availability of such information. Marketing companies for instance often need pictures in their daily activities and these companies would certainly benefit from more structured tagging systems, where tags can be grouped in clusters identifying the different meanings they have. A structured approach to tag representation would definitely improve searching, browsing, and retrieving pictures [5]. Therefore, we aim to improve the search and exploration of



We found **1,306,764 results** matching Apple.

Green apple by Hugo Provoste
130 comments 419 faves 14 notes
Tagged with light, shadow, macro, verde ...
Taken on May 30, 2007, uploaded May 30, 2007
See more of Hugo Provoste's photos, or visit his profile.

My retired Apple IIGS - but still working! by nk@flickr
16 comments 8 faves 1 note
Tagged with appleiigs, computer, monitor, appleii ...
Uploaded July 16, 2005
See more of nk@flickr's photos, or visit his profile.

Figure 1: Search results for 'Apple' at Flickr.

tag spaces by coping with syntactic variations, typographical mistakes, synonyms, homonyms, and related tags.

As a solution to the previous problem, we define the Semantic Tag Clustering Search (STCS) framework (a preliminary version is available in [16]), which consists of three parts. The first part deals with syntactic variations, whereas the second part is concerned with deriving semantic clusters. Last, the framework consists of a part where one can search in tag spaces by using search methods utilizing these clusters. We consider non-hierarchical clusters, where we select the method proposed by [15], as differently than other methods, this algorithm allows tags to appear in multiple clusters, enabling easy detection of different contexts for tags. Also, we propose an adjusted method that improves the clustering results. Finally, we devise a search method, of which the results are compared with a case without knowledge about the semantic clusters or syntactic variation clusters. We have made available an implementation of the STCS framework in the form of a Web application called XploreFlickr.com [17]. This Web application enables users to compare the results obtained from different clustering and search techniques directly on a subset of the Flickr database.

This paper continues with discussing related work on syntactic variation, semantic clustering from tags, and searching tag spaces in Section 2. In Section 3 we elaborate on our framework, followed by Section 4 that discusses the implementation of the STCS framework. We evaluate our results in Section 5 and present our conclusion in Section 6.

2. RELATED WORK

This section gives a brief overview of related work on the three main tasks of the STCS framework. Subsection 2.1 discusses literature on syntactic variation, Subsection 2.2 elaborates on semantic clustering from tags, and last, Subsection 2.3 reviews methodologies to search tag spaces.

2.1 Syntactic Variations

Syntactic variations between tags form a widely studied research subject, as they represent a well-known symptom in tagging systems. In [7], the authors analyze the performance of the Levenshtein distance [11] and the Hamming distance [8]. The authors state that Levenshtein and Hamming distances provide similar results for some syntactic variation types, e.g., for typographic errors. In contrast, for variation identification based on the insertion or deletion of characters, the Levenshtein distance performs significantly better than the Hamming distance.

This does not imply that the Levenshtein distance performs well enough, as it has problems with for instance identifying variations based in the transposition of adjacent characters, although results can be improved by ignoring candidate tags with less than four characters. An example of an implementation of the Levenshtein distance measure is recent work by Specia and Motta [15]. The authors employ the Levenshtein similarity metric to group morphologically similar tags. They use a high threshold to determine both similar words and misspellings. Within each group of similar tags, one tag is selected to be the representative of the group, and the occurrences of tags in that group are replaced by their representative. As existing algorithms only aim for larger tags, in this paper we provide a solution for this problem by also considering the short tags.

2.2 Semantic Clustering

In previous approaches, the semantic symptoms are dealt with by either using a clustering technique which results in non-hierarchical clusters of tags, or a hierarchical graph of either tags or clusters of tags. There is an extensive body of literature available on tag clustering. Several measures that cluster related tags are based on co-occurrence data, e.g., Specia and Motta [15] use the cosine similarity. The authors present a complete framework where they address the syntactic variations in a tagging system, create clusters of semantically related tags, and within each cluster, identify the relationship between each tag pair. Their semantic clustering algorithm distinguishes itself from other approaches, by allowing tags to occur in multiple clusters. Given highly similar pairs of tags, the presented algorithm considers each pair as seeds forming an initial cluster, and then tries to enlarge this cluster by finding tags that are similar to both initial tags. This procedure is recursively repeated for all tags.

Specia and Motta also experiment with different metrics to calculate the similarity between pairs of vectors of co-occurrence data, including the Euclidian and Manhattan distance. However, the cosine similarity measure is reported to yield the best results. Absolute distance metrics such as the Euclidian and Manhattan distance are inappropriate, as they are more sensitive to significant variations in a few elements than little variations in a large number of elements. In the case of Flickr, we deal with a data set with little variations in a large number of elements, and thus for our research we opt for the cosine similarity.

Similar to Specia and Motta, Begelman et al. [5] also create semantic clusters of tags by using co-occurrence data. For every tag in the data set, they find the tags with the highest co-occurrence, after which a cut-off value – determined by the first and second derivative of the co-occurrence count – is used, which places the tags above this value in a graph with the co-occurrence counts as the weights of the edges. To split the clusters further, the authors use the spectral bisection algorithm [13]. Subsequently, the modularity function [12] is used for accepting or rejecting the partitioning. The algorithm then proceeds recursively on each accepted partition. The authors conclude that clustering techniques can and should be used in combination with tagging. They also argue that these techniques can improve the search and exploration in tag spaces in general.

In this paper we focus on non-hierarchical clustering, as the antithesis – hierarchical clustering – is more complex and thus more time consuming, because it first needs to build the tag hierarchy from which subsequently the clusters are deduced [14]. The amount of data that we are dealing with asks for fast clustering procedures. Current non-hierarchical clustering approaches, e.g., the algorithm proposed by Specia and Motta [15], suffer from merging issues, i.e., larger clusters merge too quickly and smaller clusters merge too slowly. In this paper, we provide a solution to this problem.

2.3 Searching Tag Spaces

There is little literature focusing primarily on the improvement of search and exploration in tag spaces by using clustering methods. Even though the previously discussed papers covering the topic of semantic clustering aim for improving search and exploration in tag spaces, none of them seem to investigate this aspect in detail, as only derived syntactic or semantic clusters are discussed.

Seven ranking algorithms for querying tag spaces are presented in [2] and [3]. These algorithms can be applicable for users, tags, and resources. As we are interested in resources, we only consider FolkRank [9], as SocialPageRank [4] and GroupMe [1] are not suited for topic related ranking. Unfortunately, FolkRank is not applicable to the Flickr tagging system, as it requires a resource to be annotated by multiple users. In Flickr this is not possible, because only one user can upload a specific picture and annotate that picture. However, this problem is only specific to Flickr, as in other systems (e.g., Delicious) it is possible to have multiple users linking to a specific resource (a Web page).

When searching in regular tag search engines, hardly any extra information about a query is returned to the user. One could think of contexts, syntactic variations, related tags, and relationships between tags. In this paper we will improve search and exploration in tag spaces by making use of the previously developed clustering techniques.

3. FRAMEWORK DESIGN

This section introduces the Semantic Tag Clustering Search (STCS), a framework for building and utilizing clusters for the browse and search activities in social tagging systems. The framework has three parts. In Subsection 3.1 we present our technique for removing syntactic variations. Subsection 3.2 gives the procedures for finding semantically related tags. In Subsection 3.3 we improve search and exploration in tag spaces using the previously defined techniques. The input data set is defined as a tuple $D = \{U, T, P, r\}$, where U, T, and P are the finite sets of users, tag IDs, and pictures, respectively, and r is the ternary relationship $r \subseteq U \times T \times P$, defining the initial annotations of the users.

3.1 Syntactic Variations

In order to facilitate syntactic variation detection and removal, we create a set $T' \subset \mathcal{P}(T)$, where $\mathcal{P}(T)$ represents the power set of T. Elements of T' represent clusters of tags where each tag occurs only in one cluster. Hence, if $X, Y \in T'$, $X \neq Y$, and $a \in X$ and $b \in Y$, this implies $a \neq b$. Subsequently, m' denotes the bijective function that indicates a label for each $x \in T'$, $m' : T' \to L$. Furthermore, for each $l \in L$ there is a $x \in T'$ such that $l \in x$, i.e., l is one of the tags in cluster x. In this context, we employ the normalized Levenshtein similarity $\widetilde{lev}_{ij}$ between tags i and j. In contrast to the absolute Levenshtein distance, this measure makes use of string sizes, and is defined as

$$\widetilde{lev}_{ij} = \frac{lev_{ij}}{\max\left(\text{length}\left(t_i\right), \text{length}\left(t_j\right)\right)} \ . \qquad (1)$$

The algorithm for the syntactic variation clustering uses an undirected graph $G = (T, E)$ as input. The set T contains elements which represent a tag id, and E is the set of weighted edges (triples (t_i, t_j, w_{ij})) representing the similarities between tags. Weight w_{ij} is calculated as a weighted average based on the normalized Levenshtein distance $\widetilde{lev}_{ij}$ and the cosine similarity between tags i and j based on co-occurrence vectors, $\cos\left(\text{vector}\left(i\right), \text{vector}\left(j\right)\right)$, i.e.,

$$\begin{aligned} w_{ij} = \quad & z_{ij} \times \left(1 - \widetilde{lev}_{ij}\right) + \left(1 - z_{ij}\right) \\ & \times \cos\left(\text{vector}\left(i\right), \text{vector}\left(j\right)\right) \ , \end{aligned} \qquad (2)$$

where

$$z_{ij} = \frac{\max\left(\text{length}\left(t_i\right), \text{length}\left(t_j\right)\right)}{\text{length}\left(t_k\right)} \in (0, 1] \ , \qquad (3)$$

with $t_k \in T, \text{length}\left(t_k\right) \geq \text{length}\left(t\right) \forall t \in T$ and $t_i, t_j \in T$. Normalized Levenshtein values are not representative for short tags, which is why the cosine value gets more weight as the maximum tag length gets shorter. The algorithm then proceeds by cutting edges that have a weight lower than a threshold β. The syntactic clusters are computed by determining the connected components in the resulting graph.

3.2 Semantic Clustering

In order to be able to cluster semantically related tags based on their meaning, we create a set T'', with $T'' \subset \mathcal{P}(L)$, which represents clusters of elements from $l \in L$. Assignment of multiple contexts per tag is possible by allowing tags to belong to multiple clusters. Subsequently, we use the cosine similarity based on co-occurrence vectors for measuring semantic relatedness. Cosine similarity between co-occurrence vectors a and b (where $a, b \in \mathbb{R}^m$, with m representing the number of tags) is denoted as $\cos(a, b)$, and is defined in the range of $[-1, 1]$. For vectors $a, b \in {\mathbb{N}_0}^m$, the range is equal to $[0, 1]$. Cosine similarities of 0 indicate no semantical relatedness, whereas values approaching 1 indicate a high semantic relatedness.

The semantic clustering algorithm as discussed here has originally been proposed by [15]. The algorithm differs from a classical clustering algorithm, because of the fact that instead of using centroids for calculating the distance between two clusters, all tags are utilized. The main advantage is that all the elements within a cluster must be similar amongst each other, instead of being similar just to the centroid. We improve the algorithm by replacing a heuristic for merging similar clusters by a disjunction of two new heuristics. We now continue with elaborating on the algorithm.

Initially, each tag is considered as a cluster. Subsequently, tags are added to an arbitrary cluster if they are sufficiently similar to that cluster, i.e., when the average cosine of a tag with respect to all elements in the cluster are larger than a threshold χ. Because many tags are similar to each other, the set of initial clusters can contain many (nearly) duplicate clusters. Hence, there is a need for cluster merging. In [15], two heuristics taken in disjunction are proposed for this purpose. The first heuristic merges two clusters if one cluster K contains the other cluster L and is denoted as

$$K \subseteq L \; . \tag{4}$$

The second heuristic checks for small differences between clusters. Whenever clusters differ within a small margin, the distinct words from the smaller cluster are added to the larger cluster, while removing the smaller cluster. A limitation of the latter heuristic is that it makes use of a constant threshold for merging clusters, which need to be optimized first, i.e., the larger clusters should not merge too quickly and the smaller clusters should not merge too slowly. To address this issue, we propose a dynamic threshold, resulting in two new heuristics. The first one considers the semantic relatedness of the difference between two clusters, whereas the second one considers the size of the difference between two clusters in combination with a dynamic threshold.

The first adapted heuristic uses the semantic relatedness of the difference between two clusters. We merge two clusters K and L, where $|K| \geq |L|$, when the average cosine $\overline{\cos}(K,L)$ is above a certain threshold δ, and thus

$$\overline{\cos}(K,L) > \delta \; , \tag{5}$$

where the average cosine is defined as

$$\overline{\cos}(K,L) = \sum_{l \in L-K} \frac{\sum_{k \in K} \frac{\cos(\text{vector}(k), \text{vector}(l))}{|K|}}{|L-K|} \; . \tag{6}$$

The second adapted heuristic takes into account the size of the difference between two clusters, combined with a dynamic threshold. We merge the clusters when the normalized difference $\eta(K,L)$ between the clusters K and L is smaller than a dynamic threshold ε, and thus

$$\eta(K,L) < \varepsilon \; , \tag{7}$$

where the normalized difference is defined as

$$\eta(K,L) = \frac{|L-K|}{|L|} \; , \tag{8}$$

and the threshold ε is defined using a parameter ϕ as

$$\varepsilon = \frac{\phi}{\sqrt{|L|}} \; . \tag{9}$$

Hence, we are able to calculate the maximum number of different elements for the small set to be merged with the big set with

$$f(|L|) = \lfloor \varepsilon \cdot |L| \rfloor = \left\lfloor \phi \cdot \sqrt{|L|} \right\rfloor \; . \tag{10}$$

Finally, the distribution of the maximum allowed difference between clusters can be tuned by means of the parameter ϕ. Thus, we have created a function that suits the clustering process better, as it is less sensitive to the size of the smaller cluster. In [15], the maximum number of different elements was proportional to the size of the smaller set.

3.3 Searching Tag Spaces

We improve the search and exploration in tag spaces by employing clusters that provide information which can be used to more precisely specify the query. Also, a search engine should be able to recognize syntactic variations and contexts of tags. The search engine of the proposed STCS framework sorts the pictures based on relevance with the query. Instead of using clusters for picture sorting, the average cosine similarity is employed. We can sort the results by defining a similarity measure between a query and a picture, and then sort the pictures based on this similarity.

We begin by defining the query q as an m dimensional row vector of tags q_i, and a picture p as an n-dimensional row vector of tags p_j, where $q = [q_1 \; \cdots \; q_m]$ and $p = [p_1 \; \cdots \; p_n]$. We can then define the function $g(q,p)$, which calculates the similarity between the query and the picture as

$$g(q,p) = \frac{1}{n} \sum_{j=1}^{n} \left(\frac{1}{m} \sum_{i=1}^{m} \cos(q_i, p_j) \right) \; . \tag{11}$$

For a given query, for each returned picture the similarity $g(q,p)$ is calculated. The results are then sorted on a descending direction, using the previously defined similarity.

An important feature of the search engine is the automatic replacement of syntactic variations by their corresponding labels. In the framework, labels are computed, which are subsequently mapped to tags. These tags are then called syntactic variations of their tag label. When a tag is not a syntactic variation, the tag label is represented by the tag itself. The search engine can utilize this information by searching for each keyword not only on the verbatim keyword, but also on all syntactic variations of the keyword. This greatly increases the recall of our search engine.

Another feature of the search engine is the ability to detect contexts. If a tag can have multiple meanings, the search engine asks the user to choose a cluster to indicate the sense that was actually meant. In this way, clusters are used as approximations of the many contexts a tag can participate in, and can potentially improve the precision of our search engine.

4. IMPLEMENTATION

The STCS framework has been implemented in a Java-based Web application, i.e., XploreFlickr.com [17], which has been developed using the Spring framework [10]. This section elaborates on the application and its features briefly in terms of data processing, syntactic clustering, semantic clustering, and search and exploration improvement. Figure 2 gives an overview of the architecture of the implementation.

The application uses a subset from the Flickr database, containing $1,683,111$ associations ($U \times T \times P$), $57,009$ users, $166,544$ pictures, and $317,657$ tags, that are collected from two non-overlapping time intervals [2008-01-14, 2008-08-01] and [2008-08-12, 2009-02-28]. The data set is subject to cleaning procedures, as pictures have many unusable tags due to the freedom of the users in setting picture tags. To address this problem, we apply a sequence of filters that

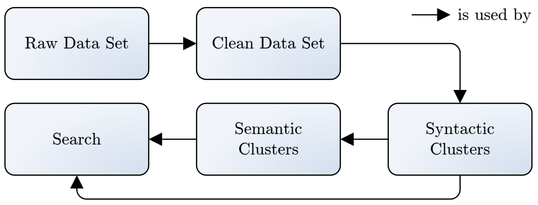


Figure 2: Overview of implementation architecture.

remove tags with unrecognizable signs, tags which are complete sentences, etc. Due to these cleaning procedures, the final data set contains 1, 231, 818 associations, 50, 986 distinct users, 147, 132 pictures, and 27, 401 tags. The syntactic clustering makes use of the full cleaned data set, whereas the rest of the application utilizes the 5, 000 most frequent tags (for performance reasons).

XploreFlickr.com implements the syntactic variation detection as elaborated on in Subsection 3.1. To this end, the application creates mappings between tag labels and possible syntactic variations of tags. For capturing semantics the non-hierarchical clustering as described in Subsection 3.2 is implemented. The Web application allows the user to enter a disjunctive query in the form of comma-separated keywords. Please note that keywords might consist of multiple words. Conjunctions are also supported through prefixing a '+' sign to a keyword. There is an auto-complete feature, as shown in Figure 3, which automatically shows keywords from all tags that start with a specific character sequence. Also, query execution is enhanced with automatic syntactic variation detection and the user is informed when this happens. Figure 4 demonstrates this variation detection for a query for self portraits.

Also, the user is presented the context of a query, to enable query refinement. Whenever more than one context is detected (e.g., recall that 'Apple' could refer to either a brand or fruit), the user is asked to select one of the contexts of the tag, or to select all contexts, as depicted in Figure 5. After the selection the search results consist of images related to that specific meaning. Figure 6(a) shows part of the first page of returned results for the fruit context, whereas Figure 6(b) depicts a small part of the search results when selecting the brand 'Apple'.

The implementation of the STCS framework supports several query mechanisms, i.e., 'Dummy', 'NHC', and 'NHC STCS'. Here, 'Dummy' refers to a search strategy that does

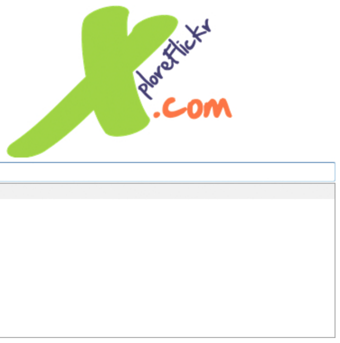
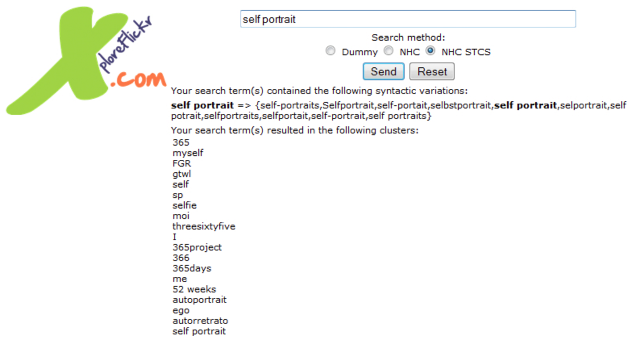

ap
Apple
applique
APlusPhoto
Apocalypse
apron
apples
Apartment
aperture
ape
april

Figure 3: Auto-completion at XploreFlickr.com.


self portrait

Search method:
Dummy NHC [✓ NHC STCS]

Your search term(s) contained the following syntactic variations:
self portrait => {self-portraits,Selfportrait,self-portait,selbstportrait,**self portrait**,selportrait,self potrait,selfportraits,selfportait,self-portrait,self portraits}
Your search term(s) resulted in the following clusters:
365
myself
FGR
gtwl
self
sp
selfie
moi
threesixtyfive
I
365project
366
365days
me
52 weeks
autoportrait
ego
autorretrato
self portrait

Figure 4: Syntactic variation detection for 'self portrait' at XploreFlickr.com.

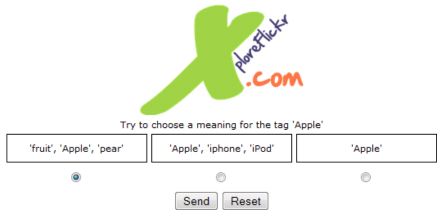
not make use of knowledge about semantic clusters or syntactic variation clusters, i.e., it simulates regular search engines that retrieve pictures using (tag) string matching. The 'NHC' option refers to the non-hierarchical clustering of [15], whereas 'NHC STCS' indicates the STCS non-hierarchical clustering algorithm. The search methods 'NHC' and 'NHC STCS' return pictures in response to a query. For each tag in the query, the methods present their associated non-hierarchical clusters (if any) in a textual format.

5. EVALUATION

This section presents experimental results of the implementation of our STCS framework. Subsection 5.1 elaborates on the results of removing syntactic variations, Subsection 5.2 discusses experimental results on creating semantic clusters, and the proposed improvements for search and exploration in tag spaces are presented in Subsection 5.3.

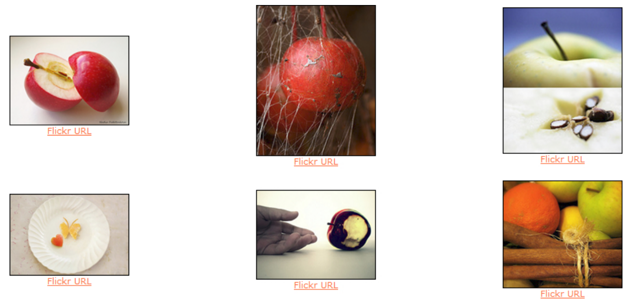
5.1 Syntactic Variations

In order to analyze the performance of the system in terms of syntactic variations detection, we define a test set S that contains 200 randomly chosen tag combinations ($S \subset T \times T$) that are subject to the normalized Levenshtein value and that have been classified as syntactic variations of each other by the STCS framework. The distributions of the tag length for the test set and the original data set of 27, 401 tags (see Section 4) are approximately the same. In our experiments, the normalized Levenshtein values for all tag combinations is calculated by means of the SimMetrics Java library [6], where we apply a threshold value β of 0.62 for cutting edges, which is determined by result evaluation using a hill climbing procedure. After manually checking these tags on correctness, we identify 10 mistakes that are produced by the framework, resulting in a syntactic error rate of 5%.

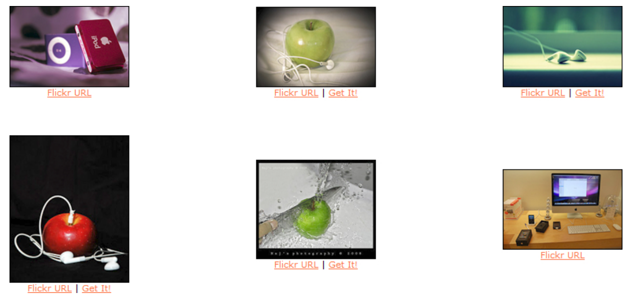

Try to choose a meaning for the tag 'Apple'

'fruit', 'Apple', 'pear'	'Apple', 'iphone', 'iPod'	'Apple'
●	○	○

Figure 5: Context selection for 'Apple' at XploreFlickr.com.



(a) Fruit context



(b) Brand context

Figure 6: Contexts for 'Apple' at XploreFlickr.com.

5.2 Semantic Clustering

We create a test set in order to be able to analyze the semantic clustering process by estimating the semantic error rate. This set contains 100 randomly chosen clusters, of which the size distribution is similar to that of all clusters. Before clustering can take place, we first optimize the threshold values for the non-hierarchial clustering algorithm using hill climbing procedures. Our analysis focuses on three thresholds, i.e., χ, δ, and ε. The first threshold, χ, determines whether or not a tag is added to a cluster during the initial cluster creation, and is set to 0.8. The second threshold, δ, defines the minimum average cosine similarity when merging two sets of which the smaller set has elements that the larger set does not contain, and is set to 0.7. As parameters for the function that defines the dynamic threshold ε we use $\phi = 0.8$, as this yields optimal results in our conducted experiments.

In our experiments, after generating 100 random clusters, we obtain 458 tags. For each cluster we count the number of misplaced tags, i.e., elements that should have been placed in another cluster. We encounter 44 misplaced tags and thus the error rate is 9.6%. Most of the misplaced tags are part of clusters containing over 20 tags. In general, the algorithm finds many relevant clusters, such as {rainy, Rain, wet, raining} and {iPod, iphone, mac}. Furthermore, a lot of clusters are found that contain tags that are translations of concepts in different languages, e.g., {springtime, primavera}.

We repeat the process of test set creation (cluster generation) and error rate calculation with the original, unadapted algorithm proposed by Specia and Motta [15]. For the constant threshold ε in the original algorithm, we determine an optimal value of 0.2 through a hill climbing procedure. Generating 100 random clusters using the benchmark algorithm results in a total number of tags of 467, including 61 misplaced tags. This results in an error rate of approximately 13.1%, and thus the STCS framework semantic clustering algorithm outperforms on this randomly selected data set the algorithm proposed in [15]. Another observation is that our algorithm produces a total of 739 clusters, whereas the original algorithm produces 421 clusters. Therefore, our algorithm discovers more clusters and thus relationships between tags. The distribution of the cluster sizes is approximately the same for both methods. A summary of the results of the experiments elaborated on in this section is given in Table 1. Here, for both clustering methods error rates are computed, together with the number of clusters found and average, minimum, and maximum cluster size.

5.3 Searching Tag Spaces

For evaluation purposes, we compare the cluster-driven search engines 'NHC' and 'NHC STCS' with the 'Dummy' search engine. This comparison is based on the precision of the first 24 results of an arbitrary query (p@24). We do not take into account other measures such as accuracy, sensitivity, and specificity, as they are difficult to derive because they require knowledge on false and true negatives, which is hard to obtain in large data sets. The sorting algorithms of the implementation are tested by randomly selecting 300 tags from the data set, and subsequently removing meaningless tags, leaving 107 useful tags for queries.

A one-tailed z-test on the results shows that the cluster-driven search engines perform significantly better (with an α of 0.0001) than the 'Dummy' search engine with respect to precision. Furthermore, we observe that the number of results for the cluster-driven search engines is much larger than the number of results for the 'Dummy' search engine. This is the result of the syntactic variation detection feature of the cluster-driven engines.

With respect to context recognition, we observe that the 'NHC' algorithm finds 214 tags that occur in at least two different clusters (implying different contexts), whereas the adapted version, 'NHC STCS' retrieves 368 tags that have at least two contexts (from the original 27,401 tags). Thus, our experiments show that the method for non-hierarchical clusters that is proposed in this paper finds more contexts than the original approach.

When searching using regular tag search engines, no extra information about a query – such as syntactic variations and contexts – is returned to the user. Our cluster-driven search methods, however, implement query refinement features. We now continue with evaluating search improvement by these additional features, using the same previous 107 tags. We queried the cluster-driven search engines using these tags, resulting in 214 queries. The initial results for both methods are the same, except for the queries where contexts are detected. When using cluster information to narrow the query, the results are different for the two methods, average precision 0.86 for 'NHC' and 0.88 for 'NHC STSC'. We perform z-tests (with an α of 0.0001) to evaluate precision improvement compared to 'Dummy', and we observe that the precision for 'NHC' and 'NHC STCS' algorithms significantly improves when compared to the 'Dummy' search method.

Table 1: Performance of non-hierarchical semantic clustering methods.

Technique	Error rate	Number of clusters	Avg. size	Min. size	Max. size
STCS framework	9.6%	739	4.4	2	67
Specia and Motta	13.1%	421	4.6	2	63

6. CONCLUSIONS

In this paper, we have proposed the Semantic Tag Clustering Search (STCS) framework for building and utilizing semantic clusters based on information retrieved from a social tagging system, i.e., Flickr. The framework has three core tasks: removing syntactic variations, creating semantic clusters, and utilizing obtained clusters to improve search and exploration of tag spaces. The STCS framework has been implemented and tested in a Web application, called XploreFlickr.com.

For the syntactic clustering process we have proposed a measure based on the normalized Levenshtein value, combined with the cosine value based on co-occurrence vectors. Results show that the framework obtains an error rate for syntactic clustering of 5%. For semantic clustering we compared the non-hierarchical clustering method proposed by Specia and Motta [15] to an adapted version that implements improved cluster merging heuristics. The STCS non-hierarchical clustering algorithm has a lower error rate than the original algorithm and produces finer-grained results. With respect to a traditional search engine, searching tag spaces using STCS retrieves more relevant results and achieves a higher precision.

As future work, we would like to improve the process of removing syntactic variations by using two ideas. First, we want to take into account abbreviations, as the Levenshtein distance is not useful for these. Second, we would like to experiment with variable cost Levenshtein distances, which associate different weights to edit operations depending on update characters and their location. For semantic clustering, the condition to merge is a disjunction of two heuristics and an earlier proposed heuristic. One could also consider other combinations, like a conjunction of the two new heuristics, to investigate whether this further improves the clustering process.

7. REFERENCES

[1] F. Abel. GroupMe!, 2010. From: `http://groupme.org/`.

[2] F. Abel, N. Henze, and D. Krause. Analyzing Ranking Algorithms in Folksonomy Systems. Technical Report, L3S Research Center, 2008. From: `http://groupme.org/papers/techreport-ranking-in-folksonomies.pdf`.

[3] F. Abel, N. Henze, and D. Krause. Ranking in Folksonomy Systems: Can Context Help? In J. G. Shanahan, S. Amer-Yahia, I. Manolescu, Y. Zhang, D. A. Evans, A. Kolcz, K.-S. Choi, and A. Chowdhury, editors, *17th ACM Conference on Information and Knowledge Management (CIKM 2008)*, pages 1429–1430. ACM, 2008.

[4] S. Bao, G. Xue, X. Wu, Y. Yu, B. Fei, and Z. Su. Optimizing Web Search Using Social Annotations. In C. L. Williamson, M. E. Zurko, P. F. Patel-Schneider, and P. J. Shenoy, editors, *16th World Wide Web Conference (WWW 2007)*, pages 501–510. ACM, 2007.

[5] G. Begelman, P. Keller, and F. Smadja. Automated Tag Clustering: Improving Search and Exploration in the Tag Space. In L. A. Carr, D. C. D. Roure, A. Iyengar, C. A. Goble, and M. Dahlin, editors, *15th World Wide Web Conference (WWW 2006)*, pages 22–26. ACM, 2006.

[6] S. Chapman. SimMetrics Package, 2010. From: `http://sourceforge.net/projects/simmetrics/`.

[7] F. Echarte, J. J. Astrain, A. Córdoba, and J. Villadangos. Pattern Matching Techniques to Identify Syntactic Variations of Tags in Folksonomies. In M. D. Lytras, J. M. Carroll, E. Damiani, and R. D. Tennyson, editors, *1st World Summit on The Knowledge Society (WSKS 2008)*, volume 5288 of *LNCS*, pages 557–564. Springer, 2008.

[8] R. W. Hamming. Error Detecting and Error Correcting Codes. *Bell System Technical Journal*, 26(2):147–160, 1950.

[9] A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. FolkRank: A Ranking Algorithm for Folksonomies. In *Workshop on Information Retrieval (FGIR 2006)*, 2006. From: `http://www.uni-hildesheim.de/de/8168.htm#id22532`.

[10] R. Johnson. Spring - Platform for Java Entrerprise Applications, 2010. From: `http://www.springsource.org/`.

[11] V. I. Levenshtein. Binary Codes Capable of Correction Deletions, Insertions, and Reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[12] M. E. J. Newman and M. Girvan. Finding and Evaluating Community Structure in Networks. *Physical Review E*, 69(2):1–15, 2004.

[13] A. Pothen, H. D. Simon, and K.-P. Liou. Partitioning Sparse Matrices with Eigenvectors of Graphs. *SIAM Journal on Matrix Analysis and Applications*, 11(3):430–452, 1990.

[14] P. Schmitz. Inducing Ontology from Flickr Tags. In L. A. Carr, D. C. D. Roure, A. Iyengar, C. A. Goble, and M. Dahlin, editors, *15th World Wide Web Conference (WWW 2006)*, pages 206–209. ACM, 2006.

[15] L. Specia and E. Motta. Integrating Folksonomies with the Semantic Web. In E. Franconi, M. Kifer, and W. May, editors, *4th European Semantic Web Conference (ESWC 2007)*, volume 4519 of *LNCS*, pages 503–517. Springer, 2007.

[16] J. W. van Dam, D. Vandic, F. Hogenboom, and F. Frasincar. Searching and Browsing Tag Spaces Using the Semantic Tag Clustering Search Framework. In I. Harris and M.-L. Shyu, editors, *4th IEEE International Conference on Semantic Computing (ICSC 2010)*, pages 436–439. IEEE Computer Society, 2010.

[17] J. W. van Dam, D. Vandic, F. Hogenboom, and F. Frasincar. XploreFlickr.com, 2010. From: `http://www.xploreflickr.com/`.