

# Finding Meaning on YouTube: Tag Recommendation and Category Discovery

George Toderici    Hrishikesh Aradhye    Marius Paşca    Luciano Sbaiz    Jay Yagnik  
Google Inc.  
1600 Amphitheatre Parkway  
Mountain View, California 94043  
gtoderici@google.com

## Abstract

We present a system that automatically recommends tags for YouTube videos solely based on their audiovisual content. We also propose a novel framework for unsupervised discovery of video categories that exploits knowledge mined from the World-Wide Web text documents/searches. First, video content to tag association is learned by training classifiers that map audiovisual content-based features from millions of videos on YouTube.com to existing uploader-supplied tags for these videos. When a new video is uploaded, the labels provided by these classifiers are used to automatically suggest tags deemed relevant to the video. Our system has learned a vocabulary of over 20,000 tags. Secondly, we mined large volumes of Web pages and search queries to discover a set of possible text entity categories and a set of associated is-A relationships that map individual text entities to categories. Finally, we apply these is-A relationships mined from web text on the tags learned from audiovisual content of videos to automatically synthesize a reliable set of categories most relevant to videos – along with a mechanism to predict these categories for new uploads. We then present rigorous rating studies that establish that: (a) the average relevance of tags automatically recommended by our system matches the average relevance of the uploader-supplied tags at the same or better coverage and (b) the average precision@K of video categories discovered by our system is 70% with  $K=5$ .

## 1. Introduction

The documents indexed by the larger Web search engines are usually accessible via textual search interfaces, which allow Web users to enter terms or phrases in natural language in order to retrieve the documents that are automatically deemed to best match the queries. The matching can be roughly approximated as a comparison between terms from the textual query on one hand and terms from the documents on the other. Non-textual Web search (such

as image search or video search) relies heavily or solely on textual content surrounding or otherwise explicitly accompanying the images or videos. While such an approach is inexpensive, practical and often highly effective (e.g., in Web image search), it can reliably retrieve only those items that are accompanied by sufficient, relatively high-quality textual content, and leads to false matches when the textual content is irrelevant or spam.

Popular video-sharing websites such as YouTube or MetaCafe ask the uploaders to specify textual content to accompany each of the thousands of videos that are collectively uploaded daily. Although uploaders can provide textual information in several forms including a title, a description and a set of free-form tags, there still are *meta-data deserts* – videos whose titles are too short or whose descriptions and tags are either brief or missing. Moreover, the true semantics of the uploaded video is often only partly captured by the uploader-supplied text. For example, a YouTube video including scenes from a user’s jet ski adventure might mention the user’s name in the description and the terms *waverunner*, *big jump*, *superman* and *porn* in the tags (in a typical scenario). Yet the knowledge that the video is about water sports is not encompassed in any of these uploader-supplied tags. As such, serving such a video to a user searching for water sports would be impossible.

In this context, the main contributions of the method described in this paper are twofold. First, it assigns additional tags to existing, already-uploaded videos. Building upon the work of Aradhye *et al.*[1], it trains classifiers based on the correspondence between content (audiovisual) features and uploader-supplied metadata, avoiding any additional manual annotation of videos. The audiovisual feature set used here is much richer and was computed on a far larger corpus of videos, leading to a much larger vocabulary of learned tags. Feedback from human raters indicates that the average relevance of the automatically-assigned tags, which increases the scope of queries for which the individual videos can be retrieved, is comparable to that of tags assigned manually by the users who upload the videos. Sec-

ond, videos deemed similar based on these recommended tags are identified and grouped together into sets labeled with textual categories (e.g., videos corresponding to *water sports*), where the categories do not have to appear in the textual description or the individual video or even of other similar videos. For this purpose, we constructed a repository of categorized instances (e.g., pairs such as *<jet ski, water sports>* or *<honda s2000, sports cars>*) from the unstructured text on the Web to bridge the gap between the user-assigned tags, the content features and the larger semantic space of categories to which user queries may refer, and to which videos may belong. The categories simultaneously fit the videos into a larger, dynamic, text-derived conceptual hierarchy, and add a useful level of generality (e.g., from *jet ski* to *water sports*). This can be directly exploited during video search, e.g., as the added categories can be matched against the input queries, allowing for an expanded set of videos to be ranked and potentially retrieved per query, and also changing the relative ranking of the videos. Furthermore, the categories enable new browsing options. Experimental results indicate that 70% of the videos are assigned to meaningful text-derived categories, thus confirming the usefulness of the first (to our knowledge) attempt to bring together content-based features derived from non-textual media and lexical knowledge derived from text.

## 2. Related Work

**Tag-to-Tag Recommendations:** Tagging has become a popular way to organize content on the Web in order to simplify access to documents. Several websites offer the possibility to tag singers/bands [11] or images [20]. A detailed description of different forms of tagging is presented in [15] including an analysis of user tagging behavior for images. Traditionally, the activity of tagging was reserved to an authority, such as a librarian. With the advent of the Web, every user is able to tag resources. Golder *et al.* discuss the difficulties of tagging systems in this scenario [6]. In particular, they point out the problems of polysemy (multiple words can be used to represent the same concept), synonymy (a word may have multiple meanings) and the different “basic level” of abstraction in describing a resource. In order to assist the users with the tedious task of tagging, several researchers addressed the problem of tag recommendation. Sigurbjörnsson and Zwol presented a tag recommendation method for the Flickr photo sharing website [24]. The system is based on the photo meta-data and uses a user provided tag list to determine additional tags based on tag co-occurrence statistics. Since the co-occurrences are computed on the whole photo collection, the method is based on the collective knowledge available within the collection. Similarly Kern *et al.* study the problem of extending folksonomies, i.e. collaboratively created sets of metadata, with

additional metadata [10] and they apply the method to recommend tags for Flickr images.

**Collaborative Tagging:** The use of collective knowledge is further developed in collaborative tagging systems. In fact, in the case of sites like del.icio.us and last.fm, users are allowed to tag all resources (this is known as *free-for-all tagging*) and the system can exploit the coherence among users in order to recommend tags. Marinho *et al.* present the problem of collaborative tag recommendation in the more general context of recommender systems [14]. They deduce algorithms that are based on the user-tag and the user-resource profiles and they apply them to data taken from last.fm. Byde *et al.* apply a similar method based on the user-resource profile similarity and propose a method for tag recommendation for del.icio.us [3]. YouTube allows users to tag only the videos that they have uploaded. Therefore, collaborative tagging is not directly possible. The system that we propose is based on the collective knowledge, similar to the approach in [24] but analyzing the video content instead of the metadata.

**Image Annotation:** In the context of content-based multimedia information retrieval, researchers use visual features to search large image collections [12]. In particular, image annotation aims at associating labels to images in order to allow query-by-text retrieval. An elegant method to solve the problem consists in using a generative model that describes the joint occurrences of labels, features, and latent concepts. Li *et al.* propose a system of this type that is able to classify the scene, segment each object, and annotate the image with a list of tags [13]. Ulges *et al.* apply a generative model to build a system for tag recommendation of videos [28]. The results corresponding to a certain query can be ranked according to some confidence scores associated to the words that form the query. In the case of generative models, the value of the joint probability of visual features and labels are used to compute the scores [9]. Grangier *et al.* propose a discriminative ranking model to solve the query-by-text problem, without solving an intermediate annotation problem [7]. In contrast with these approaches, we apply a method that is completely data-driven. This means that the set of possible labels and the family of the joint probabilities of labels and features is not determined a priori. This is an important feature for the heterogeneous corpus that we consider. Moreover, the method is perfectly scalable and adaptive. For instance, if a new tag becomes popular, it is possible to train the system to add it to the list of the possible recommended labels without retraining the whole recommendation system.

**Mining of Categorized Instances:** Existing methods for extracting categorized instances from text acquire sets of instances that are either unlabeled (e.g., [21]) or associated with a category (e.g., [8, 2, 19]). The repository of categorized instances described in this paper is larger than simi-

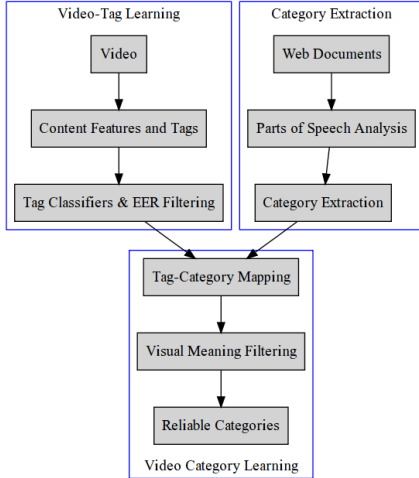


Figure 1: Overview of the proposed method for video category discovery.

lar repositories extracted from unstructured text as part of recent work. In particular, the number of useful extracted categories (e.g., categories associated with 10 instances or more) is at least one order of magnitude larger than in the repositories described in [19] and [26].

### 3. Approach

In Section 3.1 we describe the features that were extracted from each video. Next, in Section 3.2, we present a method for learning tags associated to video content. Section 3.3 summarizes the collection of text categories from the Web, while Section 3.4 presents the matching of the tags with the categories learned from text, the creation of video-specific category classifiers, and the rejection of the categories that cannot be adequately described by our features. An overview of the method is depicted in Figure 1.

#### 3.1. Feature Extraction

To be able to process a very large number of videos feature extraction has to be fast. This requirement has guided us toward implementing not necessarily the most reliable video features, but a fast set. We process both the audio signal as well as the video frames. On a typical 1-CPU workstation, videos are processed at 45 frames per second, after resizing the frames to 160x120.

**Multiscale Decomposition:** Most of our features are treated as time series over the length of the video. In case of features with multiple dimensions, we treat each dimension independently as described in [5]. We apply an 8-level 1D Haar wavelet transform on each time series. At each scale and for each of the H and L filters we compute the following moments: mean, standard deviation, min and max. This allows us to represent an arbitrarily long 1D signal with fixed

set of coefficients. The signals are sampled at 0.3 seconds, unless otherwise specified, leading to scales ranging from 0.3 seconds to 76.8 seconds.

**Audio Spectrogram and Volume:** For every audio frame we compute a 32-bin audio spectrogram. In addition, we compute the volume of the audio stream, which is represented as a single floating point value. Since the audio frequency is different from the multiscale sampling frequency, the multiscale filter performs averaging in order to achieve its 0.3 second sampling goal.

**Video Features:** We employ a shot segmentation algorithm such as [22] to compute a binary signal where a frame with a shot change was represented as 1 and the rest as zeros. The maximum value is computed for the entire sampling interval before computing the multiscale decomposition. In addition, we compute a rough global motion estimate, computed as the cosine distance between color histograms of nearby frames.

**Global Image-Based Features:** An important background feature is an 8x8 Hue-Saturation histogram. This provides the ability to get a good estimate on how colors vary over time in the video, and act as a relatively strong contextual prior for the classifiers, when combined with local features. Another important feature for a large number of video categories is the output of a face detector. We compute several statistics based on it: the ratio of the largest face to the area of the image, number of faces, and various statistics based on the skin pixels. In addition, we compute a sparse texton histogram over a vocabulary learned from images. All these are treated as individual 1D signals and the multiscale decomposition is applied to them. The final feature vector for this category has 8840 individual values. A more comprehensive description of these features is given by Rowley *et al.* [23].

**Local Features:** We employ a custom local descriptor [27, 16] which consists of Gabor responses at different orientations, spatial scales, and offsets with respect to the interest point. In total, we use 4 orientations and 27 scale-offset combinations. These are computed at interest points detected using the local maxima and minima of a Laplacian of Gaussian. This descriptor is computed only on the intensity values of the image. The color information is discarded. Once the descriptor is computed, we use a codebook and map it into a histogram. We built the 20K-word codebook using hierarchical *k-means* as proposed by Nister [17]. Since the extraction progress is a relatively expensive operation, we only perform this operation every 0.9 seconds. The histogram is normalized with respect to the number of processed frames in the video. The histogram is thresholded in order to remove codewords which are too infrequent since they have the potential to be noise.

**Final Feature Vector:** We concatenate all the feature categories (sparse and dense) and form the final feature vector



Automated Tags	Metadata	Explanation of Tags	Thumbnail
uccello, fauna, elastica, wla, multimedia, blinking, hatching, praying mantis, textile, oiseau	birds french kissing	The tags include the words for bird in Italian and French. The video shows two birds in a cage.	
fence, warszawa, galerie, barras, docks, berlino, varanasi, erfurt, purdue, duomo, olsztyn, binghamton, krakow, wrexham, tarragona, stilts, bridgeport, southend, coney, benevento, cedar, piazza, walking down, chimes, veneza, san martin, venezia	celebracion del real jaen	The tags suggest mostly city names, and “gallery” (as in a gathering of supporters). The video is actually about a gathering of people in an urban setting.	

Table 1: Examples of videos and their associated tags, and a brief explanation of the tags.

which is used for classification. On average, per video, the number of sparse local features is 2,324 out of 20,000 possible “visual words”, each requiring an index and a floating point value pair. The number of dense features is 12,308 floating point values per video.

### 3.2. Video Tag Learning

Our method aims to learn an association between videos and meaningful words. At the core of our tag recommendation system is an algorithm that can transform audiovisual features into a set of human-understandable words. We present a tag-learning system which builds on the work introduced in our prior work [1]. The authors proposed an iterative tag learning scheme which uses forward feedback. For the sake of simplicity, we only used a single iteration in this work. The audiovisual feature set used here is much richer on a far larger corpus of videos, leading to a much larger vocabulary of learned tags.

**Setup:** When uploading a video to YouTube, the user is requested to populate some structured information (although this is not mandatory) such as title, description and a set of associated tags. The description of a video can have an arbitrarily long length, and as such, it may contain information that may or may not be related to the video. The title and tags are much more succinct, and as such they are more likely to have correct information about the content of the video. We segment the title and tags of each video independently and extract n-grams up to a length of 4 from both the title and the tags. We then construct a lookup table mapping each possible n-gram to a set of videos that contain it in their metadata. We discard the n-grams that correspond to too few or too many videos.

**Training:** We then attempt to train tag-recommender classifiers, using one classifier for each n-gram (alternatively referred to as a “tag” from this point on). We use the above formed lookup table to find the “positive” samples for this tag. We select a random subset of YouTube videos that do not contain this tag in their metadata as negative samples. For each tag, we consider a set of up to 20K training

videos with up to 50% positive examples. We split the set into a training-validation partition of 70-30%. We used an AdaBoost classifier [4] with the number of stumps linearly proportional to the number of positive training samples.

It is important to note that our method differs from the traditional learning problems on video, such as those described in the TRECVID [25] in the sense that we do not have a “constrained” number of labels. In fact, our method attempts to learn all possible labels (n-grams) that appear on the site. In addition, due to the heterogeneous nature of YouTube, the videos used vary in length from a few seconds to possibly over one hour and a half in the case of presentations. In spite of this variation, our only assumption is that the target label (or tag) to be learned can be associated with the entirety of each video.

### 3.3. Categorized Instances from the Web

**Validation:** Once the training process is completed, each classifier’s performance is evaluated on the validation partition. We discard all classifiers that have an equal-error rate (EER) larger than 30%. The intuition behind choosing a relatively large EER is that many videos on YouTube are not labeled properly by the uploaders. Therefore, it is possible that the classifier trigger on videos that do not present the corresponding n-gram/tag in the metadata field. Table 1 depicts two videos and the tags for which the classifiers’ outputs were above the decision threshold. The thresholds were selected such that tags will be suggested for 80% of the videos.

**Category Extraction:** Similarly to [18], the extraction of categories relies on patterns widely used in literature on extraction of conceptual hierarchies from text [8, 2], such as

$\langle \dots \rangle C$  [such as|including|e.g.|like]  $\mathcal{I}$  [and|,|.],

where  $\mathcal{I}$  is a potential instance and  $C$  is a potential category. For instance, in the sentence: “*Investors will also keep an eye on results from European banks such as BNP Paribas [..]*”, a potential instance is  $\mathcal{I} = \text{BNP Paribas}$  and a potential category is  $C = \text{European banks}$ .

In the patterns, the boundaries of potential categories  $C$



are simply approximated from the part-of-speech tags of the sentence words, as a base (i.e., non-recursive) noun phrase identified as a sequence of adjectives or nouns ending in a plural-form noun. In the example sentence from above, the category is *European banks*, which consists of a plural-form noun and a preceding modifier. If no such phrase is found, the pattern match is discarded. In comparison, the right boundaries of the instances  $\mathcal{I}$  in the extraction patterns are identified by checking that the sequence of words within the pattern that corresponds to the potential instance  $\mathcal{I}$  (*BNP Paribas*, in the example sentence) can be found as an entire query in query logs. During matching, all string comparisons are case-insensitive. If no such query is found, the pattern match is discarded [18].

The score  $S(\mathcal{I}, \mathcal{C})$  of a pair of an instance  $\mathcal{I}$  and a category  $\mathcal{C}$ , which determines the relative rank of the category for the instance, is computed according to

$$S(\mathcal{I}, \mathcal{C}) = \text{Size}(\{\text{Pattern}(\mathcal{I}, \mathcal{C})\})^2 \times \text{Freq}(\mathcal{I}, \mathcal{C}). \quad (1)$$

Thus, a category  $\mathcal{C}$  is deemed more relevant for an instance  $\mathcal{I}$  if the size of the set of extraction patterns  $\{\text{Pattern}(\mathcal{I}, \mathcal{C})\}$  that acquire the pair is higher, and the original frequency-based score of  $\mathcal{C}$  is higher.

**Dataset:** The repository extracted from a sample of 100 million documents in English contains hundreds of thousands of categories each associated with at least 10 instances. Comparatively, a previously published repository of categorized instances extracted from a similarly-sized Web document collection [19], after a weighted intersection of pairs extracted with patterns and clusters of distributionally similar phrases, contains a total of 9,080 categories associated with instances. Subsequent extensions of the repository, using data derived from tables within Web documents, increase instance coverage, but not the number of categories [26].

### 3.4. Transferring Web Categories to Video

Since some of the video tags are instances, the extracted repository of categorized instances can bridge the gap between tags available in videos, and categories not available in videos but present in the repository. We now propose a new method to transfer the categories extracted from the Web to videos. The categories acquired from the Web are a superset of the categories in which videos can be classified. More specifically, the classifiers that are learned for the tags are a small subset of the set of all possible n-grams found on the Web. This implies that there exists a large set of categories that are simply implausible to find in the videos.

**Filtering:** In order to identify a smaller, more useful set of text-derived categories that can be assigned to the available video tags, the ranked lists of top 10 categories (e.g., *cars*, *closest competitors*, *models*, *vehicles*), if any, are retrieved from the repository of categorized instances, for each tag (e.g., *honda s2000*). Given a tag, the relative ranking of the

retrieved class labels is determined by the scoring formula described earlier. Categories retrieved for fewer than 5 tags are discarded as too specific (e.g., *niche vehicles*) and therefore unreliable, whereas categories associated with many instances in the repository of categorized instances are discarded as too general (e.g., *topics* and *factors*) and therefore relatively less useful.

The filtering process removes the most obvious “bad” categories, but it still leaves those categories which use n-grams that were learned from video, but which have a different meaning than what one would expect to find on the Web. As an example, on YouTube there are many videos labeled with city names, such as *paris* or *budapest*. This leads us to expect that the classifiers which are learned for these words are in fact some kind of outdoor/city classifiers. In practice, however, this is not the case. The AdaBoost classifier tries to learn the best possible match between videos and tags, but has no concept of meaning associated to tags. For this particular example, many of the city names have videos that contain that city name, but which are filmed in bars, night clubs, and other entertainment venues, usually by means of cellphones or other low-quality recording devices, and this is the class of videos which the classifier learns.

**Meaning-Based Filtering:** The same text term can have a different predominant meanings in the contexts of Web text documents and YouTube videos. To alleviate this problem, we need to ensure that the category in which a n-gram appears is “consistent”. For a category to be consistent the associated classifiers need to “agree” by having scores above a threshold for the same videos (the threshold is determined by setting a coverage goal of 80%). For example, this excludes categories which would encompass both city names and related monuments, as the classifiers that are learned for monuments produce vastly different scores from those which are learned for night clubs.

In order to analyze the cluster consistency potential, we compute the Median Absolute Deviation for each category (MAD). For a given video sample, we compute the individual classifier scores, then when a category has a classifier that outputs a value that is high enough, we assume that the category has a hit. Whenever a category has a hit, we compute the MAD of the scores from all the classifiers within the category. All categories with an average validation MAD over 0.15 are discarded. Table 2 depicts some of the retained categories, and some of the discarded categories. To the best of our knowledge, the idea of discarding semantic categories where the visual classifiers for sub-categories don’t give similar responses has not been considered in published literature in this area.

**Category Classifier:** The final category classifier score is simply the arithmetic mean of the component classifiers’ scores. We preferred this type of merging due to the consistency which we impose on the categories: to have a small

Retained Categories	car manufacturers, animes, guitarists, water sports, dogs, rapper, classical composers
Discarded Categories	legumes, smaller communities, application servers, northern provinces, professional designations, prestigious awards, offenders

Table 2: Top reliable and top unreliable categories as selected by our method.

Category	Components
arcade games	air hockey, cvs2, dance competition, foosball, gauntlet, neogeo, pinball, sfa3, street fighter, street fighter alpha, virtua fighter
cars	bmw, bugatti, fwd, honda s2000, hyundai, imports, indica, lancer evo, lowriders, mdx, mkii, nissan altima, outlander, palio, prelude, renault, saloon, saxo, sentra
dances	balkan, balletto, diablos, fandangos, folk dance, gaita, garba, hustle, jingle, mambo, merengue, moonwalk, orishas, polonaise, radha, reels, salsa, sevillanas, wcs

Table 3: Sample learned categories and a randomly chosen set of supporting classifiers

intra-classifier difference. Table 3 lists three categories and a subset of their selected classifier names.

## 4. Experimental Results

In the video-tag association learning stage we considered a group of approximately 53M videos split into training and validation. In addition, we used one 1M as a “hold-out” set, and it represents the set on which we ran the human evaluation.

**Relevance of Recommended Tags:** During the training time, due to the fact that the features are relatively large, and the machines which we ran our experiments on had a limited amount of memory available, for words that have over  $10K$  occurrences, we randomly sample  $10K$  which are used for training. For the negative class we randomly sample from videos which do not have the positive word in their tags or title. On this set, the video learning algorithm successfully trained 22,421 models that had a validation error of less than 30% EER.

Taking into account the large number of classifiers that are trained, it is important to assess their quality. In order to achieve this we conducted the following experiment: Independent third-party raters were asked to watch a video and rate if a supplied tag was one of these four choices: *off topic*, *somewhat relevant*, *relevant*, or *useful*, corresponding to a numeric relevance score of 0, 0.33, 0.66, and 1. These scores were aggregated over 500 randomly chosen YouTube videos by a total of 1500 third-party raters (3 raters evaluated each video). To establish a baseline, we first used uploader-supplied tags from YouTube.com in this relevance rating framework. The average relevance in this case was 0.43. Note that the raters had no knowledge about the purpose of the experiment or the source of tags (uploader-

provided or machine generated).

The accuracy of recommended tags depends on the classifier threshold. A higher threshold yields a higher relevance, but lower coverage. We chose a threshold of 0.94. We estimate that 100% of videos will have at least one recommended tag when used with our vocabulary of approximately  $20K$  tags. The average relevance of recommended tags generated by our algorithm was 0.30. We estimate that the average number of unique uploader-supplied tags and n-grams per video is about 18, whereas the same number jumps to about 55 with tags recommended by our algorithm.

A number of cases with low relevance were due to over-specialized classifiers. An example is when the recommended tag for a music video by Mariah Carey was *whitney houston*. Given the close resemblance of the music styles of these two popular female artists, the high score for the tag is not surprising, but leads to a relevance of 0 in human ratings. Given the nature of low level audiovisual features used in this work, it is not possible to get high precision artist identifiers. To eliminate this problem, we manually remapped all proper nouns in our vocabulary of tags used for testing to their respective immediate superconcepts. For example, when a classifier for the tag *whitney houston* fires, the tag *diva* is recommended instead. An experiment using the tag recommendation pipeline with this lookup step in the same rating framework led to an average relevance of 0.42.

We thus demonstrate that the tags automatically recommended by our algorithm closely match the average relevance of uploader-supplied tags, albeit with a one-time step of manual remapping of proper nouns. We are currently investigating linguistic/Web-driven automatic methods to achieve this remapping.

**Impact on Browsing Relevance:** The tags associated to the classifiers supported 3690 categories learned from the Web (after pruning). We further computed the cluster homogeneity measure described in the Section 3.4. After thresholding, the number of categories that were homogeneous enough was 235. Table 2 lists the top and bottom categories, as defined by our mean-MAD measure.

The categories which were too general, such as *offenders* were removed at this stage. Categories that had a very specific topic, i.e., *rappers* or *classical composers* were the most homogeneous. A few categories that are general yet too specific for the features and classifiers that we employ are *gm vehicles*. Although the classifiers and features are able to detect videos that contain cars, they are not able to differentiate between the various car manufacturers. An inter-category homogeneity measure would help remove such occurrences, while selecting the most general term for describing the related categories. In this paper, however, we limited at raw categories without merging them.

Table 3 lists some categories and a subset of the classi-

fiers which support them. From a linguistic standpoint, the classifiers chosen for each category seem reasonable most of the time. In order to evaluate the performance of the category classifiers, we applied them to a set of one million videos which were not used for training or validation. Figure 2 depicts the top five videos for various categories, on videos from this set.

In a browsing/discovery scenario, users are presented the categories which the algorithm has learned. When choosing such a category, they are presented with the top-scoring videos in the desired category. In order to evaluate this, we took the top five videos in each learned category and asked human raters to evaluate the correctness of the classification. The raters were presented with a video and a label. A total of 1175 videos were considered for the test, but only 1123 of them were available at the time of the experiment. The total number of raters was 1175, and each evaluated three video-category pairs. After aggregating the results, we found that for 785 (70%) of these videos the categories were deemed relevant. As noted earlier, we expected to find some categories which would be off-topic due to their high specificity.

## 5. Conclusion

In this paper we describe a method for automatic tag recommendation for YouTube uploads. In addition, we present a framework for combining categorical information from the Web with visual tag information learned from video. The proposed method discovers meaningful video categories by using video tags. The tags were evaluated in a human experiment and their immediate-superconcept meanings were roughly at about the same relevance level as the user-supplied tags. The categories discovered by the unsupervised method were demonstrated to improve the ability of users to browse on YouTube. In summary, our results indicate that: (a) the average relevance of tags automatically recommended by our system matches the average relevance of the uploader-supplied tags at the same or better coverage and (b) the average precision@K of video categories discovered by our system is 70% with K=5.

## References

- [1] H. Aradhye, G. Toderici, and J. Yagnik. Video2text: Learning to annotate video content. In *Proc. ICDM Workshop on Internet Multimedia Mining*, pages 144–151, 2009. 1, 4
- [2] M. Banko et al. Open information extraction from the Web. In *Proc. IJCAI*, pages 2670–2676, Hyderabad, India, 2007. 2, 4
- [3] A. Byde et al. Personalized tag recommendations via tagging and content-based similarity metrics. In *Proc. ICWSM*, Boulder, CO, USA, 2007. 2
- [4] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proc. ECCLT*, pages 23–37, London, UK, 1995. 4
- [5] U. Gargi and J. Yagnik. Solving the label-resolution problem in supervised video content classification. In *ACM Multimedia Information Retrieval*, pages 276–282, 2008. 3
- [6] S. A. Golder and B. A. Huberman. The structure of collaborative tagging systems. <http://www.hpl.hp.com/research/idl/papers/tags/tags.pdf>, 2006. 2
- [7] D. Grangier and S. Bengio. A discriminative kernel-based approach to rank images from text queries. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(8):1371–1384, 2008. 2
- [8] M. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proc. of COLING*, pages 539–545, Nantes, France, 1992. 2, 4
- [9] J. Jeon, V. Lavrenko, and R. Manmatha. Automatic image annotation and retrieval using cross-media relevance models. In *Proc. ACM SIGIR*, pages 119–126, New York, NY, USA, 2003. 2
- [10] R. Kern et al. Extending folksonomies for image tagging. In *Proc. Workshop on Image Anal. for Multimedia Interact. Services*, pages 126–129, Washington, DC, USA, 2008. 2
- [11] last.fm. <http://www.last.fm>. 2
- [12] M. S. Lew et al. Content-based multimedia information retrieval: State of the art and challenges. *ACM Trans. Multimedia Comput. Commun. Appl.*, 2(1):1–19, 2006. 2
- [13] L.-J. Li et al. Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. In *Proc. CVPR*, 2009. 2
- [14] L. B. Marinho and L. Schmidt-Thieme. Collaborative tag recommendations. In *Studies in Classif., Data Anal., and Knowledge Org.*, volume VIII, pages 533–540. 2008. 2
- [15] C. Marlow et al. HT06, tagging paper, taxonomy, Flickr, academic article, to read. In *Proc. HYPERTEXT*, pages 31–40, New York, NY, USA, 2006. 2
- [16] H. Neven, G. Rose, and W. G. Macready. Image recognition with an adiabatic quantum computer I. Mapping to quadratic unconstrained binary optimization, 2008. 3
- [17] D. Nistér and H. Stewénus. Scalable recognition with a vocabulary tree. In *Proc. CVPR*, volume 2, pages 2161–2168, 2006. 3
- [18] M. Paşca. Turning Web text and search queries into factual knowledge: Hierarchical class attribute extraction. In *Proc. AAAI*, pages 1225–1230, Chicago, Illinois, 2008. 4, 5
- [19] M. Paşca and B. Van Durme. Weakly-supervised acquisition of open-domain classes and class attributes from web documents and query logs. In *Proc. ACL*, pages 19–27, Columbus, Ohio, 2008. 2, 3, 5
- [20] Panoramio. <http://www.panoramio.com>. 2
- [21] M. Pennacchiotti and P. Pantel. Entity extraction via ensemble semantics. In *Proc. EMNLP*, pages 238–247, Singapore, 2009. 2
- [22] J.-C. Ren et al. Determination of shot boundary in MPEG videos for TRECVID 2007. In *TREC Video Retrieval Evaluation Online Proceedings*, 2007. 3
- [23] H. A. Rowley, Y. Jing, and S. Baluja. Large scale image-based adult-content filtering. In *VISAPP (1)*, pages 290–296, 2006. 3



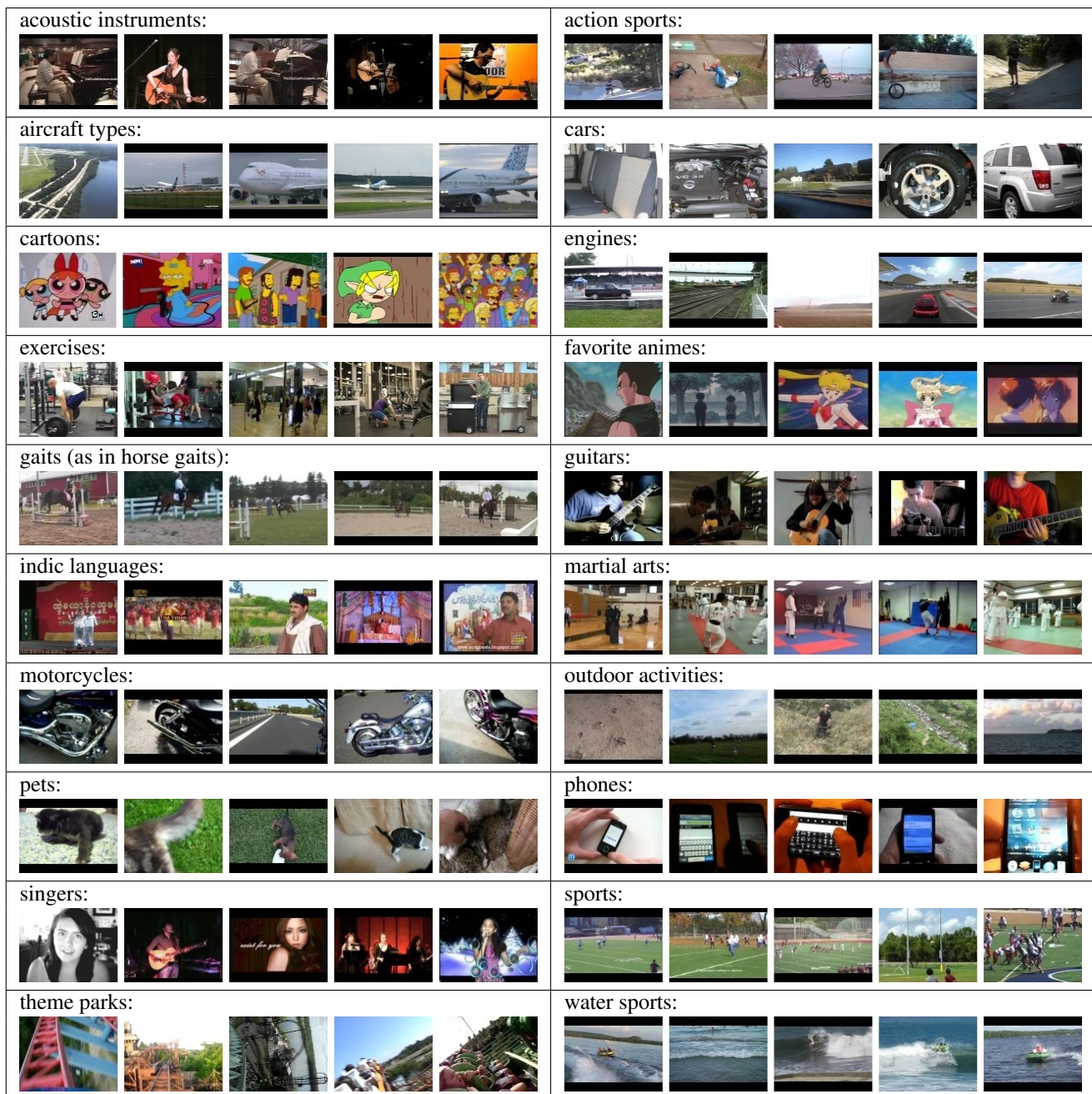


Figure 2: Example categories and the top 5 scoring videos: acoustic instruments, action sports, aircraft types, cars, cartoons, exercises, favorite animes, gaits (horses), guitars, indic languages, martial arts, motorcycles, outdoor activities, pets, phones, singers, sports, theme parks, water sports.

- [24] B. Sigurbjörnsson and R. van Zwol. Flickr tag recommendation based on collective knowledge. In *Proc. WWW*, pages 327–336, New York, NY, USA, 2008. 2
- [25] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and trecvid. In *MIR '06: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, pages 321–330, New York, NY, USA, 2006. ACM Press. 4
- [26] P. Talukdar et al. Weakly-supervised acquisition of labeled class instances using graph random walks. In *Proc. EMNLP*, pages 582–590, Honolulu, Hawaii, 2008. 3, 5
- [27] E. Tola, V. Lepetit, and P. Fua. A fast local descriptor for dense matching. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR-2008)*, 2008. 3
- [28] A. Ulges et al. A system that learns to tag videos by watching YouTube. In *Proc. ICVS*, pages 415–424, Santorini, Greece, 2008. 2