Web-scale Image Annotation

Jiakai Liu[1], Rong Hu[1,2], Meihong Wang[1,3]. Yi Wang[1], and Edward Y. Chang[1]

[1] Google Beijing Research, Beijing, 100084, China
[2] Massachusetts Institute of Technology, Cambridge, MA 02139, USA
[3] Zhejiang University, Hangzhou, 310058, China
jiakai@google.com; hurrong@mit.edu; {meihong,wyi,edchang}@google.com

Abstract. In this paper, we describe our experiments using Latent Dirichlet Allocation (LDA) to model images containing both perceptual features and words. To build a large-scale image tagging system, we distribute the computation of LDA parameters using MapReduce. Empirical study shows that our scalable LDA supports image annotation both effectively and efficiently.

Key words: Web-scale Image Annotation, distributed Gibbs sampling, Latent Dirichlet Allocation, MapReduce

1 Introduction

Although the content-based paradigm of image retrieval has been researched for over a decade, it has not been widely regarded as a success. We believe that two reasons have prevented user adoption of content-based image retrieval (CBIR). First, the involved technologies, including feature extraction, indexing, and query processing still require substantial advancements. Second, keyword search is the preferred method for users to specify a query. To support keyword search of images, effective image annotation is critical. In this work, we address the problem of image annotation.

Image annotation is traditionally treated as a machine learning problem, which consists of two major components: feature extraction and mapping features to semantic labels. Feature extraction obtains useful signals from images. Signals are then mapped to keywords via a machine learning algorithm. In this paper, we first present our overall annotation framework. We then focus discussion on our learning infrastructure: Latent Dirichlet Allocation (LDA). Our work differs from traditional methods in two respects. First, our framework considers both perceptual features and user logs in a synergistic manner. Second, our LDA implementation runs on parallel machines and can deal with data size in the order of millions. Our preliminary study in this paper shows our framework to be promising.

2 Related Work

From the point of view of learning algorithms, current automatic image annotation techniques can be divided roughly into two types: *supervised* and *unsupervised*. The supervised approach views annotation as classification, where each label word is an independent class. The simplest method is to train a series of binary classifiers, one for each label word, in a one-vs-all way [1] (where the positive class consists of images related to the label word and the negative class contains everything else). This technique has been applied to images in limited domains: to detect faces [2], to distinguish cities from landscapes [3], and so forth. More sophisticated methods of this type focus on multi-label classification [4], where labels are correlated classes, and multiple instance learning, where the training data are positive and negative bags of examples.

Compared to the supervised approach, the unsupervised approach is more scalable in the number of classes or concepts it can handle. This approach frequently employs probabilistic models to explain the co-occurence relationship between image features and semantic labels. Many models for image annotation are borrowed from the field of language and text document processing. Duygulu et al. [5] adopted a machine translation model (MT) to translate image blobs into label words. Jeon et al. [6] used a cross-media relevance model (CMRM), which assumes label words and blobs are conditionally independent given an image. These early works inspired several other variations such as Continuous-Space Relevance Model (CRM) [7], Multiple Bernoulli Relevance Model (MBRM) [8], etc. At the same time, latent space models from text processing, such as Latent Semantic Analysis (LSA), Probablistic Latent Semantic Analysis (PLSA), Non-negative Matrix Factorization (NMF), and variants of Latent Dirichlet Allocation (LDA) have all been successfully applied to image annotation [9–11]. However, although researchers have experimented with many extensions of LDA, such as Correspondence-LDA and MoM-LDA, these experiments were limited to small training sets (less than $10k$ documents and less than $1k$ unique vocabulary terms), due to the lack of parallel implementations. In this work, we use distributed Gibbs sampling of LDA implemented in the MapReduce environment to learn the latent topic distributions and the joint distribution of words and topics.

In terms of image processing, we can divide existing methods into two groups: region-based [5, 6, 10, 8], and global [12, 9, 13]. The region-based approach either segments images into "blobs" using unsupervised segmentation algorithms such as Blobworld [14], JSEG [15], and Normalized-cuts [16], or partitions images into many equal-sized rectangular blocks. Low-level feature vectors are then extracted from each region and converted into words in the visual vocabulary. Image are modeled as bags of blobs and textual words to learn the co-occurrence relationship between the two modalities. Compared to the global approach, region-based methods are more difficult to perform and evaluate for two reasons. First, they require a lot of manual labor. Second, segmentation is a resource-intensive task and may not produce blobs with semantic meanings. Whereas traditional annotation methods use only perceptual features, our framework leverages textual signals (discussed more shortly) to augment the inadequacy of perceptual features.

For evaluation datasets, a popular set for image annotation is the Corel $5k$ data set used by Duygulu et al. [5], where $4,500$ images are marked for training and the remaining 500 used for testing. However, the Corel dataset has three properties which make it suboptimal as a training set for web-scale image annotation. First, web images are of arbitrary sizes, qualities, and contents, while Corel images are all high quality professional photos on 50 CDs of clearly defined categories. Second, while web images are associated with many media modalities such surrounding text, links, user viewing statistics, etc., each Corel image only has 1 to 5 labels as groundtruth. Third, the $5,000$ photos in the Corel $5k$ dataset do not constitute a significant image collection. To date, the largest dataset used in annotation is the Corel $30k$ [17]. This number is still far far below the billions of available images on the Internet. For our work, all training images are Web images: from Web forums and from a snapshot of Google image search corpus. This choice of training data allows us to tap into query logs and words from forum post titles in addition to perceptual features in a synergistic manner, which prior work can at best uses a subset of these signals.

In summary, our annotation framework differentiates from the prior work in 1) using all available signals, and 2) addressing issues of scalability.

3 Image Representation

In this section we present our approaches for converting images into a suitable vector for LDA input.

3.1 Image Features

A major obstacle in content-based image analysis is feature selection. Despite myriads of features proposed over the years, none has come close to bridging the semantic gap between the low-level visual content of an image and what humans perceive to be the high-level semantic of the image. For large-scale image content analysis, global features like color histogram and texture are more efficient to compute than local descriptors. We choose the feature set proposed in [13] as the baseline, which is considered to be a competitive set of global features via several thorough studies. However, extracting only global features discards information about the spatial location and orientation of objects in images. Therefore, in addition to experiments using only global features, we also test combining both global descriptors and SIFT (scale-invariant feature transform) descriptors [18].

3.2 Visual Vocabulary

Because LDA assumes documents to be bags of words with counts sampled from multinomial distributions, real-valued global feature vectors and SIFT descriptors need to be discretized into a vocabulary of visual words before they can be used to train the model. We obtain a vocabulary for SIFT descriptors by K-means clustering, using the cluster ids as visual words. While we can also use K-means to cluster the global visual features, doing so would result in only one global visual word per image, which weights the global features too lightly. Therefore we choose to model each dimension of the global features vector as a Gaussian distribution and create a fixed number of bins per dimension (e.g. -4σ to 4σ). Each bin becomes a term in the global visual vocabulary, which has a fixed size of a few thousand. Each image therefore has a variable number of visual words corresponding to SIFT points, and a fixed number (144) of visual words corresponding to global features.

3.3 Textual Vocabulary

Real word labels for the training corpus can come from a variety of sources depending on the dataset. For the MIT LabelMe database, the labels are the manually created annotations accompanying each image. For pictures from internet forums, real words can come from both surrounding text and the title of the post, as well as from title of related posts. For the Google image search corpus, the real words are from both surrounding text and top keywords in the query logs.

4 Multi-Vocabulary LDA

Latent Dirichlet Allocation (LDA) is a language model, which, given documents as bags of words, clusters co-occurring words into topics, and represents documents by bags of topics. In this paper, we consider each web image as a document, which contains two kinds of words — (1) real words from surrounding text and query log, and (2) visual words from the discretization of perceptual features. This requires a slightly extended LDA model that supports multiple vocabularies. If we do not consider the relative importance of vocabularies, the multiple vocabulary LDA can be trained using the identical learning algorithm of LDA [19, 20]. However, as the old saying goes, "a picture is worth a thousand words", visual words and real words should have different importance. Inspired by the multi-vocabulary pLSA [21], we propose maximizing the following normalized log-likelihood function:

$$\mathcal{L} = \sum_{d=1}^{D} \left[\gamma \sum_{r} \frac{N_{dr}}{\sum_{r'} N_{dr'}} \log P(r|d) \;\; + \;\; (S-\gamma) \sum_{v} \frac{N_{dv}}{\sum_{v'} N_{dv'}} \log P(v|d) \right] \tag{1}$$

where D is the number of documents in the training corpus, r denotes a real word, v denotes a visual word, and γ is the blending factor of vocabulary importance.

We could optimize Eq. 1 using a variational EM algorithm that is similar to the one proposed in [19] but takes γ into consideration. However, variational EM has been shown to produce approximations too coarse for LDA. We thus use the collapsed Gibbs sampling algorithm [20]. Since Gibbs sampling is a discrete process—to infer the topic (latent variable) of every word in the training corpus, and then count the co-occurrences of every word-topic pair—we cannot derive an updating rule that is a closed form of γ. Instead, we design an approximate Gibbs sampling algorithm. We note that $\gamma \log P(r|d) = \log P(r|d)^{\gamma}$—the blending factor in effect scales the count of words in a document. So, restricting $\gamma \in [0, S]$, the approximated algorithm consists of two steps:

1. Given the training corpus, for each word w in a document d, scale $N_{wd} \leftarrow \gamma N_{wd}$, if w is a real word; otherwise, $N_{wd} \leftarrow (1-\gamma)N_{wd}$.
2. Invoke the Gibbs sampling algorithm for LDA to learn the scaled training corpus.

The larger the S, the more precisely the word distribution is kept. But a large S also incurs a higher cost in computation time, because the Gibbs sampling algorithm has a complexity linear in the corpus size, and step 1 scales the corpus size by a factor of S.

4.1 Scalable Training

In order to support large-scale image annotation, we adopt a distributed Gibbs sampling algorithm, AD-LDA, proposed in [22]. The basic idea is to divide the training corpus into P parts, each part is saved on one computer. Each computer executes one iteration of the Gibbs sampling algorithm [20] to update its local model using local data, and then the P local models are summed to form the global model, which is replicated to the P computers as the starting model for the next iteration.

Usually, the AD-LDA algorithm is implemented using the MPI programming model, which is flexible and highly efficient, but does not support automatic fault recovery—as long as one machine fails during the computation, all computers have to restart their tasks. This is not a problem when we use only a few tens of machines. But to support large-scale data like web images from image search, we need orders of magnitude more computers. Because the training process lasts many hours long, the probability of no machine failure is close to 0. Without automatic fault recovery, training will be forced to restart over and over again and statistically, may never end.

To overcome the shortcomings of MPI, we implement the AD-LDA algorithm in the MapReduce programming model [23], which has been supporting most web-scale computations at Google. MapReduce takes as input a set of key-value pairs and processes them to output a new set of key-value pairs. The entire input data is usually split into chunks known as "shards" and stored in distributed locations. A MapReduce computation job consists of three stages: mapping, shuffling, and reducing. The mapping and shuffling stages are programmable. To program the mapping stage, users provide three functions `Start()`, `Map()` and `Flush()` once. For every shard, a thread known as "map worker" is created to invoke `Start()` once, then `Map()` multiple times, once for each key-value pair in the shard, and finally `Flush()`. The map stage can output one or more key-value pairs, which are collected at the shuffling stage. Key-value pairs that come from different map workers but share the same key are aggregated to form what is known as a "reduce input", which uses the common key as its assigned key. Each reduce input is processed by an invocation of the `Reduce()` function, which is also provided by the user.

We model each iteration of AD-LDA by the following MapReduce job:

`Start()` loads the model updated by the previous iteration;
`Map()` updates the topic assignments of words in each training document using the model loaded during `Start()`, and records how the model should be updated according to the new topic assignments;
`Flush()` outputs the recorded update opinion.

Note that both the model and its update opinions are $V \times K$ sparse matrices, where V is the size of the training corpus vocabulary, and K is the number of topics specified by the user. `Flush()` outputs each row of the update opinion matrix with the key of the corresponding word[1]. The shuffling stage aggregates update opinion rows corresponding to a particular word coming from various map workers into a reduce input. `Reduce()` sums the rows element-wise to get the aggregated update opinion row for a word. The update opinion row should be added to the corresponding row of the old model estimated by the previous iteration. In a very restrictive MapReduce model, this has to be done in a separate MapReduce job. However, most MapReduce implementations now support multiple types of mappers. So rows of the old model can be loaded by an IdentityMapper and sent to `Reduce()`. It should be noted that we need to save the updated topic assignments for use in the next iteration. This can be done in `Map()`, because each document is processed once per iteration.

4.2 Scalable Model Selection

We have mentioned two parameters of the above LDA model: the vocabulary blending factor γ and the number of topics K. The values of these parameters can be determined by cross-validation—given any pair of $\langle \gamma, K \rangle$, we train a two-vocabulary LDA using part of the data (the training set) and then compute the perplexity given the trained model on a validation set [24]. The smaller the perplexity value, the better the model generalizes to new data. To support large datasets, we again model the computation of perplexity in MapReduce:

`Start()` loads the model;
`Map()` computes the log-likelihood $\mathcal{L}$ of the current input document given the model, and outputs a key-value pair, where the key is a constant value. This results in all map outputs being packed into one reduce input; the output value is a pair $\langle \mathcal{L}_d, N(d) \rangle$, where $N(d)$ is the length of document d;
`Reduce()` outputs the perplexity $perp = \exp\left[\sum_d \mathcal{L}_d / \sum_d N(d)\right]$.

5 Label Suggestion

Given a new image with visual words $\{w_1, w_2, w_3, ..., w_n\}$ and previously calculated $p(word|topic)$ parameters, $p(topic|doc)$ is computed for this image. After we get the topic distribution for this image, we can suggest real words as its tags. We tried two ways of label suggestion:

1. The posterior probability of real words given a new image is :

$$p(word|image) = \sum_{i=0}^{K} p(topic|image) \cdot p(word|topic). \tag{2}$$

   Words with big values of $p(word|image_{new})$ are considered to be more closely related to the new image and more suitable as labels.

[1] This paper is based on Google MapReduce implementation. Another well known implementation of the MapReduce model is Hadoop (`http://hadoop.apache.org`), which does not expose shard boundaries to programmers via `Start()` and `Flush()`.

2. As the new image has a topic distribution and each real word in the vocabulary also has a topic distribution, we can calculate the similarity between the image and the real words using the Jensen-Shannon distance:

$$D_{JS}(X||Y) = \frac{1}{2}[D_{KL}(X||M) + D_{KL}(Y||M)], \tag{3}$$

where $M = \frac{1}{2}(X + Y)$, and D_{KL} is the Kullback-Leibler divergence.

$$D_{KL}(X||Y) = \sum_{n=0}^{N} p(X = n)[\log_2 p(X = n) - \log_2 p(Y = n)]. \tag{4}$$

Words with similar topic distributions as the image are considered to have similar meanings with the new image and suggested as labels.

6 Experiments and Results

In this section, we report the performance of our annotation method on three sets of data. We start with the MIT LabelMe dataset to test our framework, since LabelMe images have accurate annotations. Then we move to Google image search corpus and generate labels from query keywords based on click rates. The experiments were done on 5 to 90 machines. The training process takes between 2 hours to 20 hours, depending on the number of topic parameters, the word count scaling factor γ, the number of machines, and the number of other jobs executed on the same machines in the cluster.

6.1 Datasets

MIT LabelMe Dataset. The LabelMe database [25] contains a collection of manually annotated digital images and is freely available for download. Anyone can upload their own images to the database and use the online annotation tool to outline and label objects in the images. The majority of the dataset are good quality photos taken in and around college campuses. We filtered from this dataset some repeated and unannotated images.

Google Image Search Static Corpus. This dataset is a snapshot of Google image search results taken over two days in 2007. It includes the top queries and up to $1,000$ images returned for each query. Associated with each image result are queries which returned that image and the number of clicks under each query. We can therefore model the image as a vector of query terms and use the number of clicks (or a function of it) as the word count. We use only English queries and restrict the real word vocabulary size to $10k$. Images which are not associated with top queries are filtered out. In total we used $30,638$ images for training.

Tianya Laiba forum images. Tianya is a popular Chinese internet forum with millions of users. Laiba is a social network that lets Tianya users connect with each other, join communities, and post messages to forums. Images from Laiba forum posts have more information than web images, such as users views and comments. Such additional data can help us learn better models. We extract $31,882$ images with surrounding text from Tianya Liaba forum. Ninety percent is used to train the LDA model and ten percent is held out for testing.

6.2 Experimental Procedures

To investigate the ability of visual words and text words to represent images independently, we train a separate LDA model using one vocabulary alone. We find in each single-vocabulary model, words are well distributed over topics and images under each topic have related semantic meanings. Then we combine these two types of words together to train a multiple-vocabulary LDA model. How to combine these two kinds of words is a crucial part. For simplicity, we use a linear comination as mentioned in Section 4, where the real word counts are weighted by a factor γ and the visual word counts are weighted by the factor $S - \gamma$. The combination of (γ, S) with a low perplexity is chosen for the following experiment.

We set the LDA topic number parameter to be 500 and run many iterations of Gibbs sampling. To evaluate how well a model fits the data, we compute the log likelihood on the training set and run until convergence.

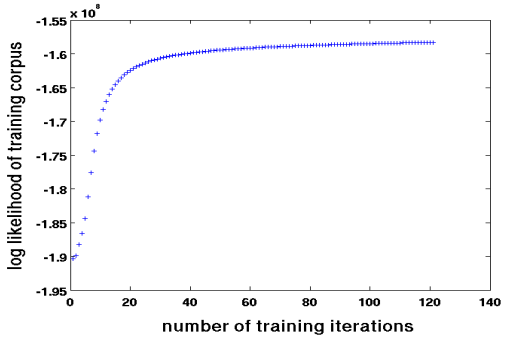


Fig. 1. Laiba training corpus log-likelihood

Figure 1 shows that the log-likelihood of the training corpus converges after about 40 iterations. To get an intuitive feel for how well the model partitions the training set into topics, we select the most "characteristic" documents and words for each topic. Documents and words are "characteristic" for a topic z_j if they have high "char" scores [26].

$$char_{i,j} = p(z_j|m_i) \cdot p(m_i|z_j) \tag{5}$$

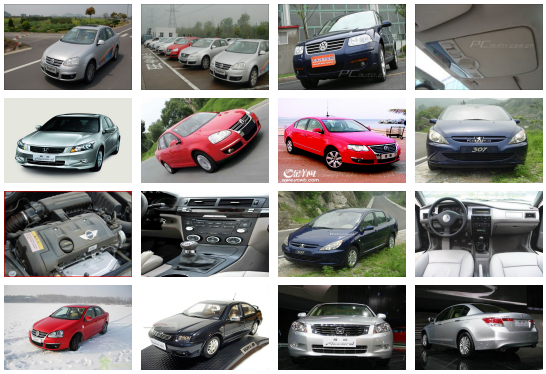
Figure 2(a) shows the top characteristic images from one topic that has an obvious semantic meaning. From these top 16 characteristic images, we can infer this topic is about cars. As Figure 2(b) shows, the top characteristic words under the same topic are also about cars[2].

6.3 Preliminary Results

We tested distributed training of LDA model on the Laiba dataset with topic number set to 500, the size of textual vocabulary at 52,767, and the size of visual vocabulary at 5,000. Figure 3(a) shows the average per-iteration training time speedup, where the speed on a single machine is set to 1. Figure 3(b) lists the training times for one iteration using different numbers of machines.

With the trained LDA model, we annotate new images using the method described in Section 5. In our experiments, some images get very meaningful annotations while others

[2] The words are translated from Chinese since Laiba is a Chinese internet forum

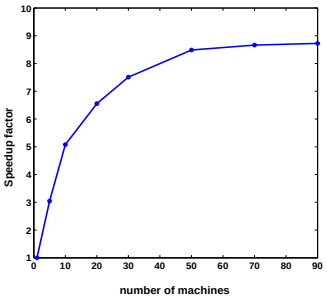


(a)

Volkswagen	Bora
speed	engine
BMW	car
transmission	fuel consumption
speaker	eighth generation

(b)

Fig. 2. (a) Top characteristic images and (b) words for topic 469



(a)

Machines	Time (sec.) per iteration	Speedup
1	1239	1
5	407	3.0
10	244	5.1
20	182	6.8
30	165	7.5
50	146	8.5
70	143	8.7
90	142	8.7

(b)

Fig. 3. (a) Plot of training time speedup and (b) actual running times

get annotations that are completely irrelevant. Irrelevant annotation might be due to low quality of these test images. Another reason is that some test images have totally different visual features compared to training images. Figures 4 and 5 show good and bad annotation results respectively.

7 Conclusion and Future Work

We have described our experiments using Latent Dirichlet Allocation (LDA) to model web images with two kinds of vocabularies: visual words from perceptual features and real words from surrounding text and query log keywords. We showed how to combine the visual words and real words and how to optimize the weight for each type of words. We also showed that the speedup of our distributed inference of LDA model parameters is significant, that our implementation is scalable and can be applied to web-scale annotation.

The image annotation process is a long pipeline with multiple stages. Effects from choices at each stage accumulate and propagate to the end. Preliminary results showed that our framework can be effective on some images, but ineffective on others. Our future work will embark on three main tasks. First, we will further optimize parallel LDA to improve speedup. Second, we will improve our data fusion model to improve accuracy. Third, we plan to devise methods to cleanse surrounding texts and user logs to obtain much more accurate predictors.

Fig. 4. Examples of good annotations

Fig. 5. Examples of bad annotations

References

1. Goh, K.S., Chang, E.Y., Cheng, K.T.: SVM binary classifier ensembles for image classification. In: CIKM '01: Proceedings of the tenth international conference on Information and knowledge management, New York, NY, USA, ACM (2001) 395–402
2. Osuna, E., Freund, R., Girosi, F.: Training Support Vector Machines: an Application to Face Detection. In: CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97), Washington, DC, USA, IEEE Computer Society (1997) 130
3. Vailaya, A., Jain, A., Zhang, H.J.: On image classification: City vs. landscape. In: CBAIVL '98: Proceedings of the IEEE Workshop on Content-Based Access of Image and Video Libraries, Washington, DC, USA, IEEE Computer Society (1998) 3
4. Change, E.Y., Goh, K.S., Sychay, G., Wu, G.: CBSA: content-based soft annotation for multimodal image retrieval using Bayes point machines. Circuits and Systems for Video Technology, IEEE Transactions on **13**(1) (Jan 2003) 26–38
5. Duygulu, P., Barnard, K., de Freitas, J.F.G., Forsyth, D.A.: Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In: ECCV '02: Proceedings of the

7th European Conference on Computer Vision-Part IV, London, UK, Springer-Verlag (2002) 97–112
6. Jeon, J., Lavrenko, V., Manmatha, R.: Automatic image annotation and retrieval using cross-media relevance models. In: SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, New York, NY, USA, ACM (2003) 119–126
7. Lavrenko, V., Manmatha, R., Jeon, J.: A model for learning the semantics of pictures. In: Advances in Neural Information Processing Systems 15. (2003)
8. Feng, S., Manmatha, R., Lavrenko, V.: Multiple Bernoulli relevance models for image and video annotation. Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on **2** (June-2 July 2004) II–1002–II–1009 Vol.2
9. Monay, F., Gatica-Perez, D.: PLSA-based image auto-annotation: constraining the latent space. In: MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia, New York, NY, USA, ACM (2004) 348–351
10. Barnard, K., Duygulu, P., Forsyth, D., Freitas, N.D., Blei, D.M., Jordan, M.I.: Matching words and pictures. Journal of Machine Learning Research **3** (2003) 1107–1135
11. Blei, D.M., Jordan, M.I.: Modeling annotated data. In: SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, New York, NY, USA, ACM (2003) 127–134
12. Carneiro, G., Chan, A.B., Moreno, P.J., Vasconcelos, N.: Supervised learning of semantic classes for image annotation and retrieval. IEEE Trans. Pattern Anal. Mach. Intell. **29**(3) (2007) 394–410
13. Tong, S., Chang, E.: Support Vector Machine active learning for image retrieval. In: MULTIMEDIA '01: Proceedings of the ninth ACM international conference on Multimedia, New York, NY, USA, ACM (2001) 107–118
14. Carson, C., Belongie, S., Greenspan, H., Malik, J.: Blobworld: Image segmentation using expectation-maximization and its application to image querying. IEEE Trans. Pattern Anal. Mach. Intell. **24**(8) (2002) 1026–1038
15. Deng, Y., b. s. Manjunath: Unsupervised segmentation of color-texture regions in images and video. IEEE Trans. Pattern Anal. Mach. Intell. **23**(8) (2001) 800–810
16. Shi, J., Malik, J.: Normalized Cuts and Image Segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8) (2000) 888–905
17. Wu, Y., Chang, E.Y., Tseng, B.L.: Multimodal metadata fusion using causal strength. In: MULTIMEDIA '05: Proceedings of the 13th annual ACM international conference on Multimedia, New York, NY, USA, ACM (2005) 872–881
18. Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. Int. J. Comput. Vision **60**(2) (2004) 91–110
19. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of Machine Learning Research **3** (2003) 993–1022
20. Griffiths, T.L., Steyvers, M.: Finding Scientific Topics. Proceeings of the National Academy of Science of U.S. (April 2004) 5228–5235
21. Cohn, D., Hofmann, T.: The missing link - a probabilistic model of document content and hypertext connectivity. In: Advances in Neural Information Processing Systems 13, Cambridge, MA, MIT Press (2001)
22. Newman, D., Asuncion, A., Smyth, P., Welling, M.: Distributed Inference for Latent Dirichlet Allocation. In: Advances in Neural Information Processing Systems 20, Cambridge, MA, MIT Press (2008) 1081–1088
23. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. In: OSDI'04: Proceedings of the 6th Symposium on Operating Systems Design & Implementation, Berkeley, CA, USA, USENIX Association (2004) 10–10
24. Heinrich, G.: Parameter estimation for text analysis. Technical report (2008)
25. Russell, B.C., Torralba, A., Murphy, K.P., Freeman, W.T.: LabelMe: A Database and Web-Based Tool for Image Annotation. Int. J. Comput. Vision **77**(1-3) (2008) 157–173
26. Cohn, D., Chang, H.: Learning to probabilistically identify authoritative documents. In: ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (2000) 167–174