Computing the Tag Genome

Jesse Vig
Department of Computer Science and Engineering
University of Minnesota
jvig@cs.umn.edu

Shilad Sen
Math, Statistics, and Computer Science Department
Macalester College
ssen@macalester.edu

John Riedl
Department of Computer Science and Engineering
University of Minnesota
riedl@cs.umn.edu

ABSTRACT

Tags help users understand a rich information space, by showing them specific text annotations for each item in the space and enabling them to search by these annotations. Often, however, users may wish to move from one item to other items that are similar overall, but that differ in key characteristics. For example, a user who loves Pulp Fiction might want to see a similar movie, but might be in a mood for a less "dark" movie. In separate work we introduce Movie Tuner, a novel interface that supports navigation from one item to nearby items along dimensions represented by tags. In the present paper we describe a data structure called the tag genome that enables this form of navigation. The tag genome encodes each item's relationship to a common set of tags by applying machine learning algorithms to user-contributed content.

ACM Classification Keywords

H.5.3 Information Interfaces and Presentation: Group and Organization Interfaces—*Collaborative computing*; H.5.2 Information Interfaces and Presentation: User Interfaces

General Terms

Design, Experimentation, Human Factors

Author Keywords

example critiquing, tagging, recommender systems

1. INTRODUCTION

Tagging systems have become increasingly popular on the Web. Users of tagging systems create free-form descriptors of music, pictures or encyclopedia articles and use these descriptors to navigate complex information spaces. In contrast to expert-designed ontologies, tags are based on the interests of the user community, tags are applied by users free of charge, and tags describe both factual and subjective aspects of items [5, 4].

Technical Report, September 10, 2010, Minneapolis, MN USA.

However, tags present challenges when used in navigation. Tagging systems lack the hierarchical structure of expert-designed taxonomies like the Dewey Decimal System [5]. Users searching for an item must specify a tag capturing their query instead of drilling down through system-specified alternatives. Studies have shown that some users find it difficult to think of tags [4].

We wish to enable a novel form of navigation that is based on tags, but that offers a fundamentally different form of navigation than traditional tagging systems. We motivate our system with a hypothetical dialogue between a movie navigation system and a user Marco:

Marco: *I'd like to watch a movie, but I'm not exactly sure what I want.*
System: *How about When Harry Met Sally, Up, or Reservoir Dogs?*
Marco: *Reservoir Dogs looks like a possibility, please tell me more.*
System: *It is a classic, non-linear, violent, crime, cult film.*
Marco: *I'm not in the mood for something quite that violent.*
System: *Then how about The Usual Suspects? It's like Reservoir Dogs, but less violent.*
Marco: *I'll take it!*

In separate work [6], we introduce *Movie Tuner*, a novel application hat enables users to navigate an information space much like Marco did. In the present paper, we show how we compute the *tag genome*, the underlying data structure that drives Movie Tuner. Informally, the tag genome describes a set of items in terms of their relationship to a common set of tags, in the same way that a biological genome describes organisms based on a common set of genes. The tag genome provides the data necessary to display the relevance of tags to items, to compare items with respect to particular tags, and to find items that are similar to a given item.

In this paper we show how to construct the tag genome by applying machine learning algorithms to user-contributed content. We begin by formally defining the tag genome. We then describe the data sets used to learn the tag genome, including a gold standard of tag relevance values and a set of features constructed from user-contributed content. Finally, we present six regression models for computing the tag genome and we evaluate these models against the gold standard.

2. THE TAG GENOME

Just as an organism is defined by a sequence of genes, an item in an information space may be defined by its relationship to a set of tags . If T is a set of tags and I is a set of items, we quantify the relationship between each item $i \in I$ and tag $t \in T$ by the *relevance* of t to i, denoted as $\mathrm{rel}(i,t)$. $\mathrm{rel}(i,t)$ measures how strongly tag t applies to item i on a continuous scale from 0 (does not apply at all) to 1 (applies very strongly). In the movie domain, for example, rel(*Reservoir Dogs*, *violent*) = 0.98, rel(*The Usual Suspects*, *violent*) = 0.65, and rel(*A Cinderella Story*, *violent*) = 0.03.

The *tag genome* for an item i is the vector of tag relevance values across all tags in T, denoted as $\mathrm{rel}(i)$. Formally,

$$\mathrm{rel}(i) = \big\langle \mathrm{rel}(i,t_1), \ldots, \mathrm{rel}(i,t_n) \big\rangle \; \forall t_k \in T$$

The tag genome has three key features that support the Movie Tuner application. First, the tag genome provides a continuous measure of tag relevance on a consistent 0-1 scale. Second, the tag genome is dense, in that it defines a relevance value for every *tag* $t \in T$, enabling comparisons between items with respect to arbitrary tags. Third, the tag genome may be used to measure similarity between items so the system can find similar items when responding to critiques.

3. THE MOVIELENS PLATFORM

We used the MovieLens[1] movie recommender as a platform for computing the tag genome. The primary purpose of MovieLens is movie recommendation: users rate movies on a scale of 1 to 5 stars and receive recommendations in return. MovieLens has been in continuous use since 1997. 186,000 users have provided a total of 17 million ratings. MovieLens also supports tagging of movies; 5,375 users have applied 31,325 distinct tags, resulting in over 246,000 total tag applications.

4. TAG RELEVANCE PREDICTION

In this section we discuss how we construct the function $\mathrm{rel}(i,t)$, which computes the relevance of tag t to item i (see Section 2). We first describe how we collect a gold-standard training set of tag relevance values. We then define features that are used for predicting tag relevance, and discuss six regression models that use these features as inputs. Finally, we evaluate each regression model against the gold standard.

4.1 Training data set

In order to train our algorithm for predicting tag relevance, we collected a gold-standard set of tag relevance values for a subset of (*item*, *tag*) pairs. To collect this data, we conducted a survey in which we asked MovieLens users to rate the relevance of tags to movies. We included all tags applied by at least 10 users, and we sampled from movies with at least 100 ratings. In each round of the survey, subjects were shown 8 pages, each displaying a single tag and 6 movies they had rated. For each tag, subjects rated the relevance of the 6 movies to the tag on a scale of 1 (does not apply at all) to 5 (applies very strongly). 676 users participated in the survey, providing relevance ratings for a total of 50,203 (*item*, *tag*) pairs. We performed a linear transform on the relevance ratings to put them on a 0-1 scale.

[1] www.movielens.org

4.2 Features

We construct features from tagging data, item ratings, and text reviews of items. Because the output variable $\mathrm{rel}(i,t)$ is defined for an item-tag pair (i,t), each feature is also defined for an item-tag pair (i,t).

Features based on tags. We constructed several features from the tags that users have applied on MovieLens:

- **tag-count:** $\text{tag-count}(i,t)$ is the number of times tag t has been applied to item i.
- **tag-applied:** $\text{tag-applied}(i,t)$ returns 1 if tag t has been applied to item i, 0 otherwise.
- **tag-lsi-sim:** Because tags are applied sparsely, we wished to take into account other tags besides t that have been applied to item i. For example, if t = *scary*, and *scary* itself has not been applied to i, but *frightening* has been applied, *scary* is likely still relevant to i. A common method for uncovering relationships between terms is to use *latent semantic indexing* (LSI) [1]. To use LSI for tags, we construct a document-term matrix with items as documents and tags as terms, and we perform singular value decomposition on this matrix. The result is a lower-dimensional representation of each document (i.e. item) in the matrix. We then express each tag as a single-term document and transform it to the same lower-dimensional space as the items. $\text{tag-lsi-sim}(i,t)$ is the cosine similarity between tag t and item i in this lower dimensional space.

Features based on ratings. We constructed two features based on movie ratings on MovieLens:

- **avg-rating:** $\text{avg-rating}(i,t)$ is the average rating for item i. This feature does not depend on t.
- **rating-sim:** This feature measures the affinity between tags and items based on rating patterns. Specifically, $\text{rating-sim}(i,t)$ equals the cosine similarity between the vector of ratings for item i and the centroid of the rating vectors of all items tagged with t (excluding i itself).

Features based on text reviews. We collected text reviews by crawling user-contributed movie reviews on the Web. We compute two features from this data:

- **text-freq:** $\text{text-freq}(i,t)$ is the frequency that tag t appears in text reviews of item i. We removed stopwords from t and stemmed t before performing this calculation. We applied a log transform to the resulting value to make the distribution more normal.
- **text-lsi-sim:** We use the same approach described above for tag-lsi-sim, but use a document-term matrix based on the frequency that tags appear in text reviews of items.

Meta-feature. We constructed one feature that is computed from other features:

- **prob-tag:** $\text{prob-tag}(i,t)$ is the estimated probability that i should be tagged with t. To compute $\text{prob-tag}(i,t)$, we solve a classification problem specific to t where the outcome variable is $\text{tag-applied}(i,t)$, which equals 1 if tag t has been applied to item i and 0 otherwise. The predictors comprise all of the non-tag features above.

  The positive training examples comprise all items that have been tagged with t. We construct a set of negative examples by randomly sampling 1000 items that have not been tagged with t. We use a subset of 1000 non-tagged items rather than all non-tagged items in order to make the problem more tractable. We then weight these negative examples so that we effectively have the same number of negative examples as positive examples.

  Once we construct the training set of positive and negative examples, we run a logistic regression classifier using all non-tag features: $\text{avg-rating}(i,t)$, $\text{rating-sim}(i,t)$, $\text{text-freq}(i,t)$, and $\text{text-lsi-sim}(i,t)$. Because logistic regression is a *well-calibrated* classifier, it produces a probability that the class is positive. $\text{prob-tag}(i,t)$ denotes this probability.

4.3 Regression model

We compared six regression models for predicting $\text{rel}(i,t)$, including 3 linear models and 3 generalized linear models [2]. In the formulas below, $X_{(i,t)}$ denotes the feature vector for item i and tag t, and comprises all feature values associated with the pair (i,t) as well as a constant term.

Linear models. In these models, $\text{rel}(i,t)$ is modeled as a linear function of feature vector $X_{(i,t)}$. We consider three variations of the linear model:

- **Single regression.** We express the problem of tag relevance prediction as a single regression:

$$\text{rel}(i,t) = X_{(i,t)} \cdot \beta$$

  In the above expression, β denotes a vector of coefficients that is of the same length as $X_{(i,t)}$. Note that β is constant across all tags and items. The advantage of this approach is that there is abundant training data to estimate the parameters, since all training data can be used for this single regression problem.

- **Separate regression per tag.** In this model, the vector of coefficient β_t is specific to each tag:

$$\text{rel}(i,t) = X_{(i,t)} \cdot \beta_t$$

  This is equivalent to solving a separate regression problem for each tag. The advantage of this model is that it captures behaviors specific to each tag. The disadvantage is that one can only use training data collected for that specific tag. Because there is less training data, the model may overfit the data.

- **Hierarchical regression by tag.** In the hierarchical regression model [2], the vector of coefficient β_t is specific to each tag. Unlike the separate regression model, however, β_t is modeled as a random variable that follows a normal distribution $N(\mu,\sigma)$ across all tags:

$$\text{rel}(i,t) = X_{(i,t)} \cdot \beta_t,$$
$$\beta_t \sim N(\mu,\sigma)$$

  This model has benefits of both the single regression and the separate regression models. Since separate parameters are learned for each tag, the model captures tag-specific behaviors. However, because the parameters are modeled to follow a prior distribution, there is less chance of overfitting the data.

Model	MAE
linear, single regression	0.237
linear, separate regression per tag	0.253
linear, hierarchical by tag	0.220
generalized linear, single regression	0.234
generalized linear, separate regression per tag	0.224
generalized linear, hierarchical by tag	0.211

Table 1: MAE for each regression model based on 10-fold cross validation against the gold standard dataset of tag relevance ratings provided by users (transformed to 0-1 scale)

Generalized linear models. The generalized linear models are the same as the linear models, except for the addition of the logit link function:

$$\text{logit}^{-1}(x) = \frac{e^x}{1+e^x}$$

The advantage of generalized linear models is that they capture nonlinear relationships between $\text{rel}(i,t)$ and feature values $X_{(i,t)}$.

- **Single regression:**

$$\text{rel}(i,t) = \text{logit}^{-1}\big(X_{(i,t)} \cdot \beta\big)$$

- **Separate regression per tag:**

$$\text{rel}(i,t) = \text{logit}^{-1}\big(X_{(i,t)} \cdot \beta_t\big)$$

- **Hierarchical regression by tag:**

$$\text{rel}(i,t) = \text{logit}^{-1}\big(X_{(i,t)} \cdot \beta_t\big),$$
$$\beta_t \sim N(\mu,\sigma)$$

4.4 Evaluation

We evaluated each of these regression models using 10-fold cross validation on the training set. We used the R programming language to solve the regression equations [3]. The primary R functions we used were lm (linear regression), lmer (linear, hierarchical), glm (generalized linear), and glmer (generalized linear, hierarchical).

As shown in Table 1, the generalized linear models outperformed the linear models, and the hierarchical models

outperformed the single regression and separate regression models. The generalized linear, hierarchical model performed best, achieving an MAE of 0.211.

5. ACKNOWLEDGMENTS

The authors thank the members of Grouplens for their feedback and assistance. This paper is funded in part by National Science Foundation grants IIS 03-24851, IIS 05-34420, IIS 09-64695, and IIS 09-64697.

6. REFERENCES

1. S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.
2. A. Gelman and J. Hill. *Data analysis using regression and multilevel hierarchical models*. Cambridge University Press, New York, 2007.
3. R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2010. ISBN 3-900051-07-0.
4. S. Sen, S. K. Lam, A. M. Rashid, D. Cosley, D. Frankowski, J. Osterhouse, F. M. Harper, and J. Riedl. tagging, communities, vocabulary, evolution. In *CSCW '06: Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work*, pages 181–190, New York, NY, USA, 2006. ACM.
5. C. Shirky. Ontology is overrated. http://www.shirky.com/writings/ontology_overrated.html, 2005. Retrieved on May 26, 2007.
6. J. Vig, S. Sen, and J. Riedl. Navigating the tag genome. In *IUI '11: Proceedings of the 15th International Conference on Intelligent User Interfaces (to appear)*, New York, NY, USA, 2011. ACM.