Image Topic Modeling: What an Image Represents

Wen-Fu Lee
Department of Computer Science
University of Wisconisn-Madison
Madison, WI
wlee256@wisc.edu

Wenjun Qiu
Department of Industrial and
Systems Engineering
University of Wisconisn-Madison
Madison, WI
wqiu8@wisc.edu

Huan Liang
Department of Industrial and
Systems Engineering
University of Wisconisn-Madison
Madison, WI
hliang47@wisc.edu

Yahn-Chung Chen
Department of Industrial and
Systems Engineering
University of Wisconisn-Madison
Madison, WI
chen666@wisc.edu

Abstract

It's an intuition that images could be clustered by topics based on objects and tags appeared in it, where Non-Negative Matrix Factorization(NNMF) and Probabilistic Latent Semantic Analysis(PLSA) were chosen as they are the most common approximation technique for topic modeling. By applying Multiplicative Update Rule(MUR), Alternating Least Square(ALS), Projected Gradient Descent(PGD), Alternative Non-negative Gradient Descent-LIKE(AND-LIKE) and PLSA on 738 imaged with the most common tags observed in sports topic image, we managed to extract meaningful topics in all algorithms and made comparison on classification performance according to ground truth labels.

1 Introduction

One of the remarkable features of human is that human can identify, understand image and then connect it to some higher level ideas or concepts accurately and immediately. By inspecting the object, position and color, human can easily come up with numerous hidden concepts related to the image which may not actually show up in the image; likewise, human can also reminiscent of some objects and image by given hints or topics. This kind of pattern somehow shows that the identification and understanding of image are correlated. Most of researches nowadays focus on image classification, some of which try to apply "understanding" to facilitate the performance of image classification.

Topic modeling is the most common approach used for discovering the hidden semantics and meaning of text, and it can also be used for clustering labeled data. Based on this property, images labelled with the objects appeared in it can be clustered and extracted the top representative labels for describing these clusters. In this paper, several topic modeling algorithms including NNMF and PLSA are implemented by ourselves using Matlab to compare their performance of exploring hidden topic among images. Since the technique of multi-label image classification has not been mature enough, the bag-of-words for tags of image were generated manually.

The report contains 5 sections. Section 1 is the introduction part, which shows the motivation for using NNMF and PLSA for image. Section 2 shows all the related work in NNMF and PLSA use for topic modeling. Section 3 introduces the preprocessing work and data labeling process. Section

4 formally introduces several algorithms based on NNMF and PLSA. Section 5 evaluates each algorithm on various evaluation metrics. Section 6 concludes the project.

2 Related Work

It's widely acknowledged that annotation to some extent provides solid support for enhancing the performance of image classification. Research conducted by Wang et. al.[1] tries to implement generative probabilistic model for simultaneous image classification and automation. Guillamet et. al.[2] applied NNMF algorithms to classify image annotation with small image patches composed of different color tags. Besides, PLSA was applied by Pliakos et. al.[3] to extract prevailing topics among large dataset of image with tags, In this report, what we focus on is basically the implementation and comparison of NNMF algorithms and PLSA.

NNMF has been a popular tool in topic modeling ever since first proposed by Paatero et. al.[4]. The main idea is to factorize a raw data matrix into two non-negative matrices and multiple algorithms have been developed to manage that, among which ALS algorithm by Paatero et. al.[4], MUR algorithm by Lee et. al.[5,6] and PGD algorithm proposed by Lin[7] are the most representative ones.

As one of the most popular methods for topic modeling, *Latent Semantic Analysis* (LSA) [8] can successfully map original data matrix to a vector space of reduced dimensionality. Based on a decomposition of the co-occurrence matrix by SVD, LSA can represent semantic relations between words and documents in terms of their proximity in the semantic space. However, although LSA has some good properties, it is limited by its weakness of dealing with polysemy and its assumption that words and documents form a joint Gaussian model [9].

PLSA [9] proposed by Hofmann defines a proper generative data model by introducing an unobserved latent class. This generative model can get rid of LSA's limitations above. To find an appropriate estimation of this model, a maximum likelihood estimation is formulated, and can be optimized by Expectation Maximization (EM) algorithm [10]. Specifically, two different inference processes of EM can be found in [11], which are for two different interpretations of PLSA, respectively. In this report, the optimization results of the second inference process of EM are adopted and implemented to achieve the estimation process.

3 Dataset Generation

3.1 Motivation

This project requires recognizing objects in image to generate multiple tags as instances. Recent researches on object recognition shows many fancy models with promising results. However, these models are still not efficient in recognizing normal objects such as stands in basketball ground, helmets for football players. The deficiency in the results, running time for processing each image, and hardware requirements of these models make it hard and inefficient to use. Thus we decide to manually label images to generate a bunch of instances for use. After finishing labelling 50 images in 9 sports topics, we find that there exists some ground-truth patterns in each topic. For example, in basketball images, there are more images having 3 players than 1 player. The reason of choosing sports images is that different sports images tend to have distinct features between each other, which helps in generating easily identified data. Instead of manually labelling each image, which is quite time-consuming and tedious, we come up with an interesting simulation method in generating a bunch of instances given the probability of each object in different types of sports.

The model requires the probability distribution of each sport as the hyper-parameter. We use the probability extracted from a small population as the whole population's probability distribution.

3.2 Generation Process

We explicitly collect 9 types of sports, which are "Basketball", "Soccer", "Football", "Baseball", "Golf", "Tennis", "Swimming", "Skydiving" and "Hockey". And there are many similarities in objects between different sports images. To generate the instances for the project use, we first need to create a "tag pool", which can be seen as the union of unique labels for all 9 sports. Through observations, we classify the objects into 8 categories, which are "Ball Types", "Tools", "Scoring Place", "Field

Table 1: Detailed tags in each category

	Tag 1	Tag 2	Tag 3	Tag 4	Tag 5	Tag 6	Tag 7
Ball Types	Basketball	Soccer	Football	Baseball	Golf	Tennis	Hocky
Tools	Gloves	Sticks	Helmet	Cap			
Scoring place	Goal	Net	Stand	Square			
Field Color	Green	White	Brown	Blue	Line		
Outdoor	Sky	Lake					
People	Audience	**1-2 P*s**	**3-10 Ps**	**$\geq$ 11 Ps**			
Motion	Kick	**w/ tool**	**w/o tool**	**Jump**			
Others	Chairs	Bags	Cheerlead				

Color", "Outdoor Feature", "People", "Motion" and "Others". We have 33 tags in total. And the tags in each category are shown in Table 1, where P* stands for Players.

Each instance would be a sparse vector with length 33 with either 0 or 1 value in a specific tag. For example, we would have a basketball instance with tags in {Baketball, Stand, Brown, Line, Audience, 1-2 Players, Throw or Swing w/o tool, Jump} having value 1 and all the rest 25 tags with value 0.

Our assumption is that most tags are independent from each other. The probability is not conditional. However, for some tags and sports, there exists some dependencies. The bold text in Table 1 shows these dependent tags. Obviously the number of players in each image is definite, thus, only one of the three types in numbers of players can be 1. And in the "Motion" category, for baseball, all these three motions are the normal motion, but it is impossible for all these three exists in one single image. This indicates in the baseball instance generation process, only one of the three motions can have value 1.

The generated probability is based on observing 50 images of each sport in total. The detailed probability distribution is in the appendix "Generalized Label". It is possible that an all-zero instance or a quite sparse instance can be generated. To avoid the situations that the dataset contains lots of useless data, we set a validation criteria for our instance, Based on the max non-zero tags in each type, we set a validation criteria to guarantee the feature vector's actual valid tags. We assume that a valid instance should contain at least 9 non-zero values in the feature vector and set this to be the criteria. The data generation procedure can be explained in the following pseudocode.

Algorithm 1 Data Generation

Require: Empty Instance Set $\mathbf{M}$ and Probability of tags in each sport $\mathbf{P}$
for $\mathbf{p}_i$ in $\mathbf{P}$ **do**
  initialize an empty set $\mathbf{S}_0$
  for j = 1:100 **do**
    instance = DATAGEN($\mathbf{p}_i$)
    if instance meet validation criteria **then**
      put the instance into $\mathbf{S}_0$
    end if
  end for
  $\mathbf{M}$.add($\mathbf{S}_0$)
end for
OUTPUT: Instance Matrix $\mathbf{M}$

4 Implementation

4.1 NNMF

Multiple algorithms of NNMF could reach our expectation, but we're interested in which algorithm will have the best performance in our problem setting. We are glad to see that it involves topics like Least Square and Gradient Descent.

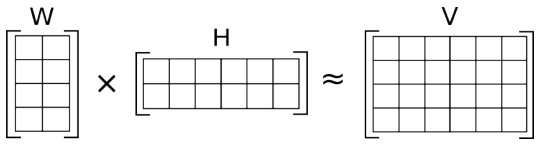

Figure 1: NNMF representation[12]

4.1.1 Multiplicative Update Rule

To evaluate whether the approximation of the original data matrix is precise enough, cost function needs to be determined. In this method, two cost functions are welcome to choose and the update rules are also different upon the choice.

For the squared Frobenius Norm loss function:

$$[W, H] = \arg\min ||V - WH||_F^2$$

We have the following update for the each entry (i,j) of matrix W and (m,n) of matrix H.

$$H_{ij} = H_{ij} \frac{(W^T V)_{ij}}{(W^T W H)_{ij} + \epsilon} ; W_{mn} = W_{mn} \frac{(V H^T)_{mn}}{(W H H^T)_{mn} + \epsilon}$$

For the Kullback-Leibler divergence measure:

$$[W, H] = \arg\min \sum_{ij} [V_{ij} \log \frac{V_{ij}}{(WH)_{ij}} - V_{ij} + (WH)_{ij}]$$

We have the following update for the each entry (i, j) of matrix W and (m, n) of matrix H.

$$H_{ij} = H_{ij} \frac{\sum_k [W_{kj} V_{kj} / ((WH)_{kj} + \epsilon)]}{\sum_l W_{li}} ; W_{mn} = W_{mn} \frac{\sum_q [H_{nq} V_{mq} / ((WH)_{mq} + \epsilon)]}{\sum_p H_{np}}$$

where ϵ is a very small positive number which prevents dividing by 0. Before iteration starts, matrix W and matrix H are initialized as a random non-negative matrices. In each iteration of both methods, we can see that one of the two matrices will get updated earlier than the other one. So we need to take into consideration the influences made by different updating sequences. As a result, we will run four different tests in total for MUR algorithm, two different updating orders for each.

4.1.2 Alternating Least Square

The main idea of the ALS algorithm is quite concise. The loss function chosen is still the squared Frobenius Norm loss function. Before iterations starts, matrix W is initialized as a random non-negative matrix.

In each iteration, first matrix W is fixed and matrix H will be calculated by solving the least square problem, which is equivalent to solving the following matrix for H. We immediately set the negative entires to 0 to keep the non-negativity.

$$W^T W H = W^T V; H = (H)_+$$

Then we fix matrix W and calculate matrix H by solving the other least square problem, which is also equivalent to solving the following matrix equation for W. The negative entries of the new matrix W also need to be set to 0 and this new matrix W will be used as the initial matrix W in the next iteration.

$$H H^T W^T = H V^T; W = (W)_+$$

It will exit the loop once the gap between loss function values of two consecutive iterations is sufficiently small.

4.1.3 Projected Gradient Descent

To better represent the gradient of the loss function, we divide our squared Frobenius Norm loss function by 2:

$$f(H, W) = \frac{1}{2}||V - WH||_F^2$$

The the gradient of $f(W, H)$ can be computed as:

$$\nabla f_H(H, W) = W^T(WH - V); \nabla f_W(H, W) = (WH - V)H^T$$

The projected Gradient Descent is the extended version of Gradient Descent, except that before the end of each iteration, all the negative entires in matrix H and calculate matrix W will be set to 0 to keep the non-negativity.

$$H = [H - \tau \nabla f_H(H, W)]_+; W = [W - \tau \nabla f_W(H, W)]_+$$

where τ is the learning rate which could be wisely chosen in order to guarantee the convergence. It will exit the loop once the gap between loss function values of two consecutive iteration is sufficiently small.

4.1.4 Alternative Non-negative Gradient Descent

Li and Liang [13] [14] has proposed a simple and natural alternating gradient descent based algorithm called Alternative Non-negative Gradient Descent (AND). It mainly contains 2 stages: Decode and Update

$$\text{Decode: } z = \phi_\alpha((\mathbf{W}^{(0)})^\dagger y)$$
$$\text{Update: } \mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} + \eta(yz^T - \mathbf{W}^{(t)}zz^T)$$

where α is a threshold parameter, and function $\phi_\alpha(x)$ is an ReLu function,

$$\phi_\alpha(x) = \begin{cases} x & \text{if } x \geq \alpha \\ 0 & \text{otherwise} \end{cases}$$

And $(\mathbf{W}^{(0)})^\dagger$ is the Moore-Penrose pesudo-inverse of $\mathbf{W}^{(0)}$, y is a single instance. By doing the update operation, each instance gets to contribute to the convergence to the feature matrix. and η is the update step size.

Algorithm 2 AND-LIKE

Require: Threshold α_0, iteration time s, instance matrix $\mathbf{Y}$
Initialize: Non-negative feature matrix $\mathbf{W}_0$
$\mathbf{W}^{(0)} \leftarrow \mathbf{W}_0$
for $j = 1 : s$ **do**
  for y in $\mathbf{Y}$ **do**
    decode:$z = \phi_\alpha((\mathbf{W}^{(0)})^\dagger y)$
    update: $\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} + \eta(yz^T - \mathbf{W}^{(t)}zz^T)$
  end for
  update $\mathbf{W}^{(0)} \leftarrow \mathbf{W}^{(T+1)}$
end for
OUTPUT: Feature Matrix $\mathbf{W} \leftarrow \mathbf{W}^{(T+1)}$
OUTPUT: Cluster Matrix $\mathbf{H} = (\mathbf{W})^\dagger * \mathbf{Y}$

There are several things to mention in implementing this algorithm. First, the initialization of this algorithm needs to be specified. It can not be too far away from the ground-truth. And the paper specified that for $\mathbf{W}_0$, the columns of it needs to contain one major feature and a small fraction of some others. However, this does not meet the assumption for this project. We are doing the unsupervised learning. A major feature can not be determined in our project practice. Thus, we decide to revise the initialization part to randomly initialize a non-negative matrix to use.

Second, the threshold value α needs to decrease exponentially with stages as specified in the paper. This is because the l2 norm error $||\mathbf{W} - \mathbf{W}* ||_2$ decease exponentially with stages, in which $\mathbf{W}*$ is the ground-truth feature matrix.

Last, the number of stages(iterations) needs to specified to guarantee the convergence. In the paper, it shows that the iteration times $s = O(D^{1/6})$. In our practice, D has been set to 9, based on this, we decide to choose s = 15. We name this algorithm as "AND-LIKE".

the pseudocode for this AND-LIKE algorithm is shown in Algorithm 2.

4.2 Probabilistic Latent Semantic Analysis (PLSA)

The PLSA is basically a statistical model called the aspect model. The aspect model is introduced with a latent class variable denoted as $z \in \{z_1, z_2, \cdots, z_K\}$. With this unobserved variable, the generative model for a tag/photo can be defined according to the following scheme.

1. Select a photo d with probability $P(d)$.
2. Pick a latent class (topic) z with probability $P(z|d)$.
3. Generate a tag w with probability $P(w|z)$.

In the application of this report, the $P(d)$ is used to denote the probability that a tag occurrence will be observed in a particular photo d. $P(z|d)$ denotes the probability distribution of an unobserved topic z given one specific photo d, and finally $P(w|z)$ denotes the conditional probability of a specific tag w given the hidden topic z. The data generation process can be translated to a joint probability model as follows.

$$P(d, w) = P(d) \sum_z P(w|z)P(z|d) = \sum_z P(w|z)P(d|z)P(z)$$

With the joint probability model above, the Log-likelihood of the whole data set (assume that all tags are generated independently) can be generated as:

$$\mathcal{L} = \sum_d \sum_w n(d, w) \log[\sum_z P(w|z)P(d|z)P(z)]$$

where $n(d, w)$ is the number of a tag w in a photo d. To obtain a maximum likelihood estimation, Expectation-Maximization (EM) is adopted. Specifically, for E-step, the posterior probability for latent variables is calculated as follows:

$$P(z|w, d) = \frac{P(w|z)P(d|z)P(z)}{\sum_z P(w|z)P(d|z)P(z)}$$

For M-step, the optimal estimation of the probabilities in the expression of the Log-likelihood of the whole data set are derived as follows:

$$P(z) = \frac{\sum_d \sum_w n(d, w)P(z|w, d)}{\sum_d \sum_w n(d, w)}$$

$$P(w|z) = \frac{\sum_d n(d, w)P(z|w, d)}{\sum_w \sum_d n(d, w)P(z|w, d)}$$

$$P(d|z) = \frac{\sum_w n(d, w)P(z|w, d)}{\sum_d \sum_w n(d, w)P(z|w, d)}$$

In our implementation, $P(z)$, $P(w|z)$, $P(d|z)$ and $P(z|d, w)$ are randomly initialized. Then, the EM process will update these four probabilities, and be repeated until all elements of $\Delta_{P(z)} < 0.00001$, which gives better performance after a range of threshold values is tested.

5 Experimental Results

5.1 Pre-setting

Either in NNMF or in PLSA, K parameter for number of topics needs to be chosen before iteration starts, which is also a tuning parameter in most topic modeling researches. However, the unsupervised

learning setting in this paper is that we assume that we already have an image dataset with 9 categories, where we don't know how much of them are with what topic. What we're interested in is how NNMF and PLSA algorithms perform on extracting meaningful topics and classifying images to its corresponding topic given the number of topics.

For the representative word, we decide to take the first 5 words to represent one cluster. The amount of words are chosen based on the experimental result. Here we show the clustering results in PLSA. From the result in Table 2, we can see that 5 tags is enough for human to identify the cluster. For example, the five tags in the second cluster can be identified as Hockey because of the "White" color field, "Stick" tool ,"Swing with tool" motion and "Helmet" and "Gloves" protection.

Table 2: Tags to represent each cluster in PLSA

Cluster	Tag 1	Tag 2	Tag 3	Tag 4	Tag 5
*	3 -10 Ps	w/o tool	Cheerlead	Line	Cap
Hocky	Helmet	Stick	Gloves	Swing w/ tool	White
Skydiving	Blue	1- 2 Ps	Bag	Sky	Helmet
Tennis	Net	Brown	1- 2 Ps	Line	Stick
Football	Gloves	Green	Sky	Helmet	w/o tool
*	Audience	Chairs	1- 2 Ps	3 -10 Ps	Line
Basketball	Jump	Stand	Brown	3 -10 Ps	Line
Soccer	Kick	Green	Sky	Jump	3 -10 Ps
Golf	Cap	Stick	1- 2 Ps	Green	Sky

5.2 Topic Interpretation

In MUR, we have 4 algorithms in total, which are using Forbenius norm loss function and update matrix $\mathbf{W}$ first (FW), using Forbenius norm loss function and update matrix $\mathbf{H}$ first (FH), using Kullback-Leibler divergence measure and update matrix $\mathbf{W}$ first (LogW), using Kullback-Leibler divergence measure and update matrix $\mathbf{H}$ first (LogH). In Figure 2 and Figure 3, The number from 0 to 8 indicates respectively the following classes: "Basketball", "Soccer", "Football", "Baseball", "Golf", "Tennis", "Swimming", "Skydiving", "Hockey".

Confusion matrix in Figure 2 can provide a good view for comparing the true and predicted label. Figure 2 showed the performance of these 6 algorithms. PLSA can perform the best classification since the sparsity of non-zero elements is relatively low, and its performance can also be verified by inspecting the main diagonal. In Confusion Matrix, the deeper the color of the main diagonal is, the better the performance of the classification is. In contrast to the other algorithms, the performance of ALS is the worst. Even if it can successfully extract interpretable topics, it tends to misclassify all images to only two topics. Comparing the 6 Confusion Matrices, we can make the conclusion that the performance of classification can follow this order: PLSA $>$ LogW $\cong$ FH $\cong$ PGD $>$ AND $>$ ALS.

One of the features of Confusion Matrix is that it can show where the true label is misclassified to. In the figure of PLSA, it shows that the image with topic, "Baseball" is easier to be misclassified to "Football", "Golf" and "Tennis". It quite makes sense because "Baseball" shares more common features with these three sports.

Figure 3 can provide a better visualization for this pattern. For PLSA, we can't extract a "make sense" cluster to represent the "Baseball" (The $\star$ sign implies the topic can not be derived from the top words), which results in the increment of elements predicted into "Tennis", "Football" and "Golf".

We could evaluate the prediction accuracy of each algorithm's percentage of images in "non-sense" topics, percentage of images in "make sense" topics but wrongly described and percentage of images in "make sense" topics and correctly described. From the last metric, AND_like stands out because of large amount of images being classified under "non-sense" topics even if it behaves better than ALS in Confusion Matrix evaluation. However, it has the second highest ratio of correctly labeled images when we set the base as images under "make sense" topics. To some extent, it avoids giving a large fraction of misclassification like ALS does when we dig deep. Apart from the two algorithms with the lowest performance, FH, PGD and LogW seem to have average and close performance,

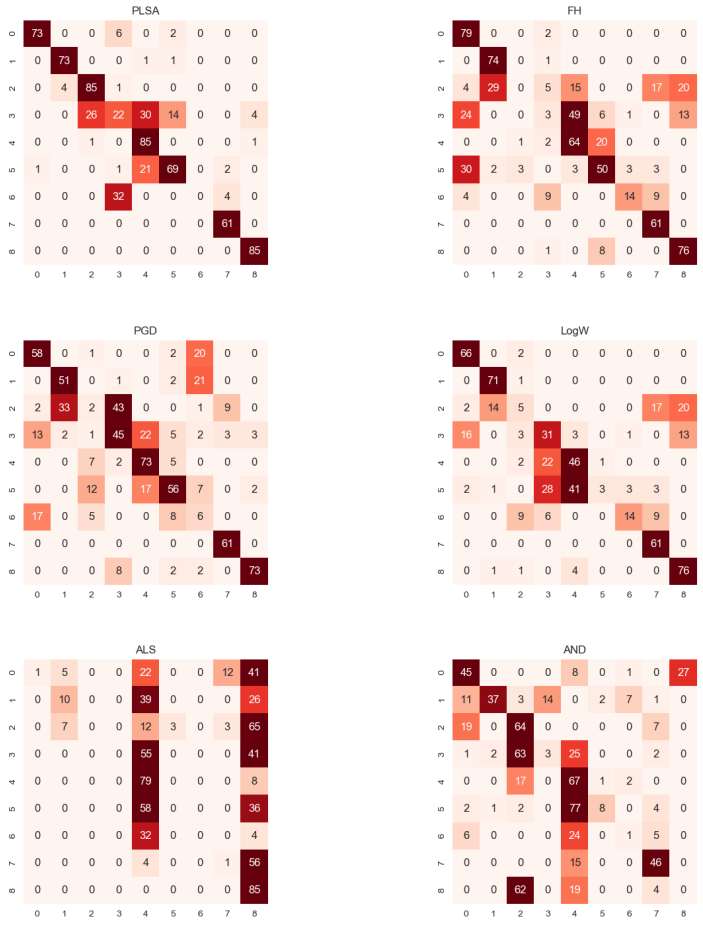

Figure 2: Confusion Matrix

high percentage in "make sense" topics and 50% to 60% correctly being classified. And due to the randomness, we can't say for sure which of them is determined to be the best among these three.

Whether from the Figure 3 and Figure 4, PLSA gives the best performance among all algorithms, where around 75% images are correctly labeled. Also, PLSA has the highest ratio of correctly labeled images no matter we set the base as all images or just images under "make sense" topics.

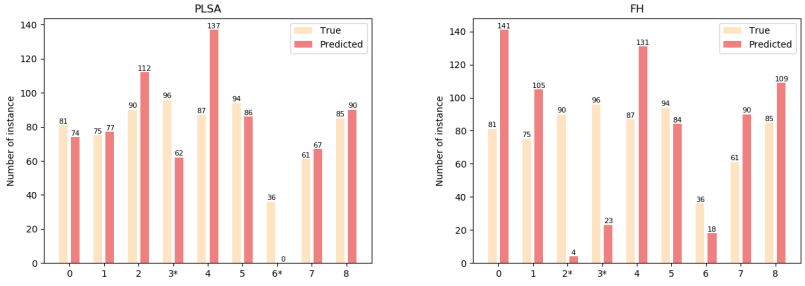

Figure 3: Actual vs. Predicted

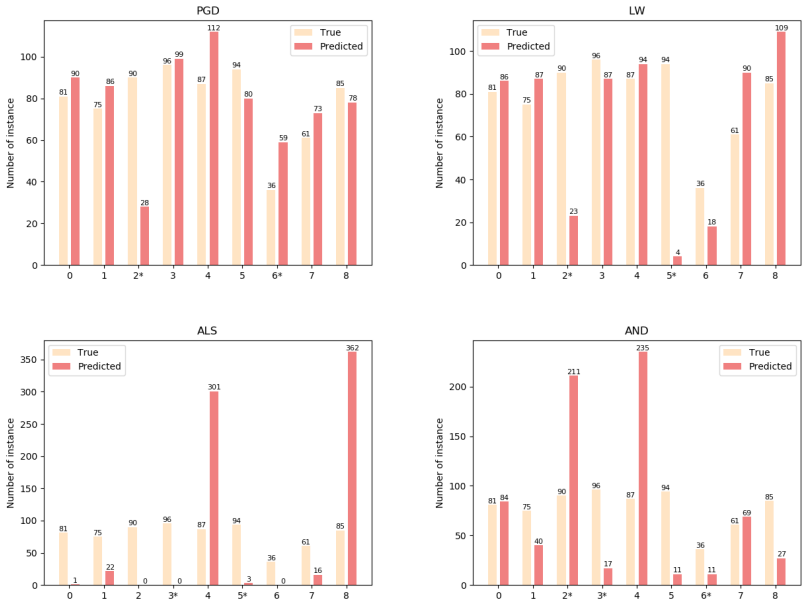

Figure 3: Actual vs. Predicted

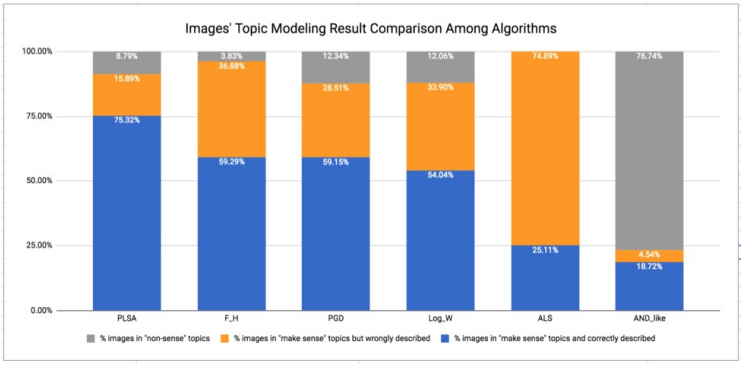

Figure 4: Algorithms comparison in various metrics

6 Conclusion and Future Work

In this project, we have mainly used Matlab to implement several algorithms based on NNMF and PLSA as unsupervised topic models to process data. The data is generated by a simulation model built by our group. Several algorithms especially PLSA show promising results in topic modeling. All the raw data files, code and results are in the source directory.

For future work, the data generation model can be compared with the manually labelled result to show its usefulness.We plan to expand our data set and expect the algorithms to have more promising performance on a larger dataset.

Acknowledgments

We would like to thank Prof. Liang for his excellent lectures, which provided us with an in-depth insight into machine learning. At the initial stage of the project, Prof. Liang gave us some really useful suggestions towards data simulation. Especially his excellent paper of Alternative Non-Negative Gradient Descent contributing to the community provided us with a new point of view to see the topic modeling problem.

References

[1] Li,L.J, Socher, R, Li Fei-Fei.(2009) IEEE Conference on Computer Vision and Pattern Recognition - 2009

[2] David Guillamet, Bemt Schiele, and Jordi Vitri.(2002) Analyzing Non-Negative Matrix Factorization for Image Classification. *Conference on Pattern Recognition ICPR*

[3] Konstantinos Pliakos-Constantine Kotropoulos. (2014) PLSA driven image annotation, classification, and tourism recommendation *2014 IEEE International Conference on Image Processing (ICIP)*

[4] P. Paatero and U. Tapper (1994). Positive Matrix Factorization: A non-negative factor model with optimal utilization of error estimates of data values. In *Environmetrics* on (**Vol. 5**, pp. 11-16)

[5] Lee, D. D., & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, **401(6755)** , 788-791.

[6] Lee, D. D., & Seung, H. S. (2001). Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems* (pp. 556-562).

[7] Lin, C. J. (2007). Projected gradient methods for nonnegative matrix factorization. *Neural computation* , **19(10)**, 2756-2779.

[8] Deerwester, S., Dumais, G.W., Furnas, S. T., Landauer, T. K., Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science, 41*, 391-407.

[9] Hofmann, T. (2001). Unsupervised Learning by Probabilistic Latent Semantic Analysis. *Machine Learning, 42*, 177-196.

[10] Dempster, A. P., Laird, N. M., Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statist. Soc. B, 39*, 1-38.

[11] Hong, L. (20). A Tutorial on Probabilistic Latent Semantic Analysis.

[12] Non negative matrix factorization: `https://wiki/Non-negative_matrix_factorization`

[13] Li, Y., Liang, Y., & Risteski, A. (2016). Recovery guarantee of non-negative matrix factorization via alternating updates. In *Advances in Neural Information Processing Systems* (pp. 4987-4995).

[14] Li, Y., & Liang, Y. (2017). Provable Alternating Gradient Descent for Non-negative Matrix Factorization with Strong Correlations. arXiv preprint arXiv:1706.04097.