Video Classification using Consensus Learning and CrowdSourcing

B.Tech. Project Phase I Report

Submitted in partial fulfillment of requirements for the degree of
Bachelor of Technology

by
Aditya Kumar Akash
Roll No : 120050046

under the guidance of
Prof. Ganesh Ramakrishnan

Department of Computer Science and Engineering
Indian Institute of Technology Bombay
Mumbai 400076, India

November, 2015

Contents

Abstract

A video conveys a lot of information about any subject than a mere picture. We currently have a huge collection of videos. So classification of the videos would help in easy and fast location of video of interest, saving time and also serving as a learning tool. In this work we work towards assigning tags to videos using crowd sourcing.

Crowd sourcing using games are one of the methods currently used to gather data in a fast and cheap way. The game interface asks player to tag the videos, for which he would be awarded points based on a scoring mechanism. We improve the existing scoring and try to build a metalearner which would be learning about tags and videos and serve as video classifier.

Meta learners based on ensemble models are popular choice for multi label multi class classifiers. Amongst different meta learners consensus learning best suits our framework. We use MLCM, a simple consensus learning model to achieve the task at hand.

Acknowledgements

My sincere thanks to Prof. Ganesh Ramakrishnan for providing valuable feedback time and again which served to influence and improve this work.

I would also like to thank the entire team: Dr. Simoni S.Shah, Post-Doctoral Fellow IIT Bombay,and my fellow team mates Depen Morwani, Saketh Vadlamudi, Shraddha Bhattad, Deepak Dilipkumar for their extended discussions and brainstorming on various issues.

Special thanks to Ashish Kulkarni, Research Scholar at IIT Bombay and Mitesh Khapra, Researcher India Research Laboratory, Bangalore, India for providing us valuable insights.

I would also like to thank Ankit Vani & Pooja Ahuja, Interns at IIT Bombay for designing the initial game framework.

Chapter 1

Introduction

1.1 Motivation

Video is a powerful tool to convey any information precisely. Looking at an action frame by frame with audio providing supporting explanation is highly informative and easier to grasp. This makes video a popular choice for learning new concepts.

Video is both appealing and intelligible to masses and that makes it an medium for instruction. Using videos farmers can stay in touch with the latest technology, success stories of other farmers around the world. A video library can easily be accessed and utilized by farmer groups using specialized mobile applications, as smart-phones have now become common. A farmer can learn from a relevant video and solve his problem without an actual expert present to guide him. The aim of this project is to facilitate such a video library where suggestions of relevant videos be made to farmers to help them solve their problem.

Present video repositories like *YouTube* have collection of videos which serve general purpose video classification. We need a classification which more suits an agriculture domain and hence it forms a closed domain. Moreover manual video collection and tagging is a tedious task. Thus we seek to use crowd sourcing as an option for video collection as well obtaining tags.

In order to attract people to participate in such a crowd sourced activity voluntarily they must be given some incentive as such a fun element. So we introduce *Video Tagging* game where users can watch the videos and provide tags for which they are given points. In this way they both enjoy the game as well as learn from it. As a result of users interacting with the game we obtain rich collection of tags which we use to predict the relevant tags of the videos. This brings in meta learner models which learn association of tags to videos which would also help in recommendation of videos and easy navigation.

1.2 Related Work

Researchers have attempted to achieve video classification. However either they only utilize the meta data of video, audio and text or require computationally expensive models to solve the problem. Some of the algorithms consider entire video as a single feature vector and approach this problem from the aspect of co-training multiple complementary views.

VideoKheti[3] attempts to present agricultural videos to farmers. However in this the main aim was not to categorize videos, but to acquire data on the user experience and the usability of multi-modal interaction system for searching and viewing of agricultural videos.

Active learning has been used to evolve personalized category catalogue for agricultural videos in [4]. However they use associative Markov networks which are computationally expensive and so inappropriate from game point of view.

VideoMule[5] combines individual classification and clustering algorithms trained on textual meta-data, audio and video through a heuristic consensus based learning. However, they have no crowd sourcing element hence they would not have evolving tag set for a given video. This is because interpretation of information present in video changes over time.

BGCM[2] deals with single label multiclass classification, such that prediction combination happens within the individual labels. We could apply this method to multi label multiclass classification by combining predictions of base models for each label, resulting in consolidated predictions for each label. Then the consolidated predictions are collected to get the final prediction. But these could not capture label correlations.

In this work, we try to achieve video classification for agricultural domain we use a crowd sourcing based approach of tag collection. We first look at how to rank the tags based on simple tf-idf scoring. We then combine them using a consensus based model. We adapt the consensus method presented in [1] to generate multiple semantic labels to the videos. This approach has less computational cost and also is based on crowd sourced data, and hence the assigned tags evolve with time.

Chapter 2

TagVideo Game

2.1 Working of the Game

The *tagVideo* game allows user to play videos and while the video plays they can start typing a tag and get a list of suggestions. The user selects from these selections and gets scored for it. The user can end the game and submit the score at any time. In the figure below we can see scores for different tags user has entered.

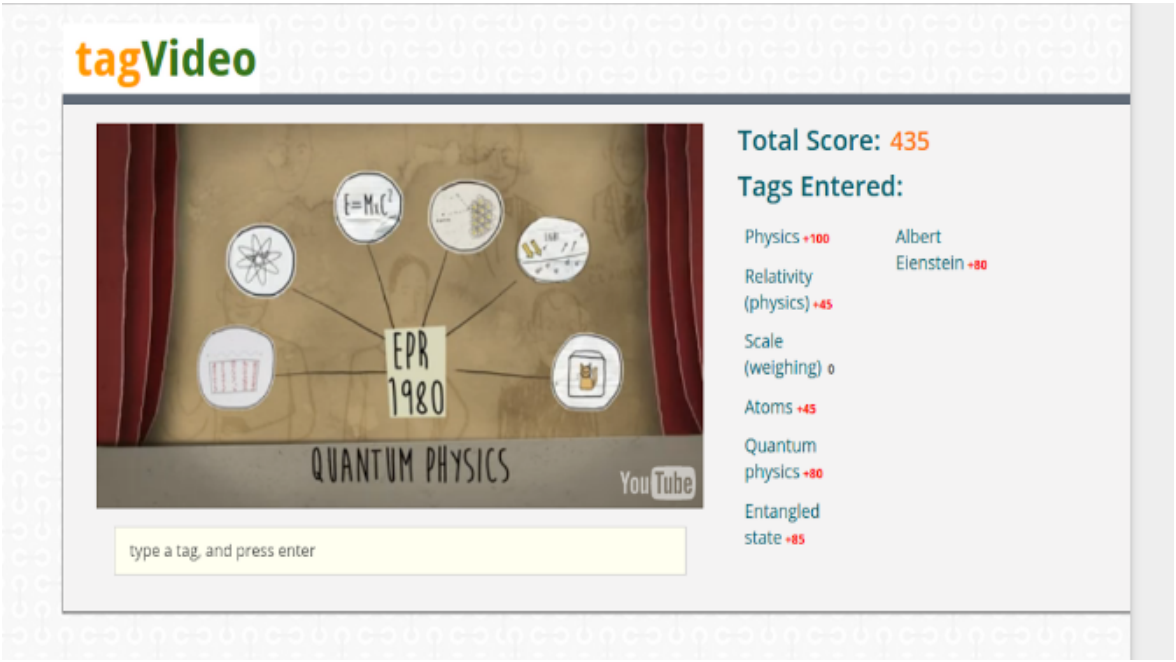

Figure 2.1: TagVideo Game in action

Based on the scores cumulative across all the videos the user is ranked against other users. This introduces competitiveness to give better tags for the videos and keeps the user attentive.

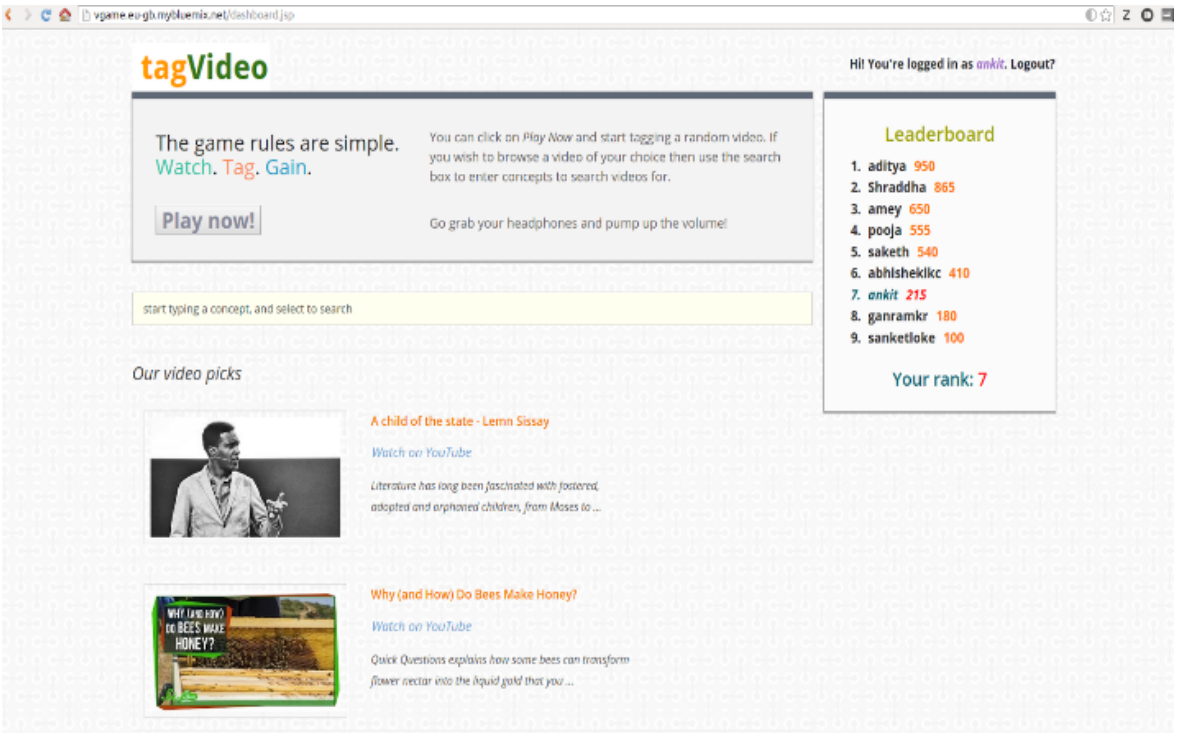

Figure 2.2: TagVideo game Dashboard

The above image shows dashboard for a given player and the global rank list on the right. User can also search for videos of his interest or click on *PlayNow* to watch a random video from the library.

This initial game framework was developed by Ankit and Puja. We next investigate the function they have used for generating the scores of the tags entered.

2.2 Existing Scoring Function

Providing scores to the tags is at the heart of this game. It keeps the player motivated to play the game as well as provide better and relevant tags in order to get better scores. The scoring function can be also viewed as an outcome of a hidden model which associates the tags to videos and generates a relevance score. Thus a good scoring function suggests a good model, and hence a good classification of videos.

All the videos are initially fed into *IBM Watson Concept Insight Corpus*. When the player starts typing a tag, we use IBM Watson Concept Insights to auto-complete and suggest possible tags. When the user selects a tag, we score based on the *IBM Watson Concept Insights relevance score* of the tag with the video. Scoring is based on how relevant a given tag is to a video.

Initially, this score is purely based on the relevance score returned by IBM Watson Concept Insights As more and more users tag a video, a *TF-IDF component* of the scoring function starts gaining more weight.We assume that tags that more people agree on are correct. Thus, if for a given video, a large number of people have given a tag, it gets a higher score. Inversely, if that tag is common in many other videos, its score for this video diminishes. This principle encourages more specific tags, inducing greater focus on the video.

Lets see the scoring function in detail.

Table 2.1: Notation of Tf-Idf Scoring

Symbol	Meaning
t	Tag
v	Video
f_v^t	Frequency of tag t in video v
tf	Term Frequency
c_v	Count of distinct tags given for video v
N	Total Number of videos in library
N_t	Number of videos in which tag t appears
idf	Inverse Document Frequency
s_{watson}	Score for a tag by IBM Watson
w_{watson}	Weight for Watson score
N_v	Count of number of users who tagged video v
tf_idf	Tf-Idf score for a tag video pair

Now we have, the term frequency which denotes in comparison to other tags how relevant is this tag for the given video,

$$tf = \frac{f_v^t}{c_v} \tag{2.1}$$

Then Inverse document frequency which denotes how specific the tag is for the video. It is inversely related to how spread the tag is across video library.

$$idf = 1 + \log \frac{N}{N_t} \tag{2.2}$$

The TF-IDF component of score,

$$tf\_Idf = tf * idf \tag{2.3}$$

The weight given to watson score is decreased with increase in number of taggers for a given video. This is done to shift on consensus of people to decide the relevant tag,

$$w_{watson} = \frac{1}{e^{c_0 * (N_v - 1)}} \tag{2.4}$$

The final score awarded to a tag for a video is given by -

$$score = s_{watson} * w_{watson} * idf * c_1 + (1 - w_{watson}) * \frac{tf\_Idf}{max_t^v(tf\_Idf)} * c_2 + c_3 \tag{2.5}$$

where max_t^v takes maximum value over tags for video v, and c_0, c_1, c_2, c_3 are constants which help to bring the score in range 1 to 100.

2.3 Analysis of Scoring Function

The scoring function shifts the score from IBM Watson score towards some kind of consensus among users with more weights to video specific tags. This is a desired property since we want that the relevance of a tag be determined by the players of the game.

Some other properties which we expect a good scoring function to follow are -

1. It should not be possible for users to co-ordinate and game the system. This means that users should not communicate and provide same tags to get high scores by virtue of term frequency.

2. Global strategy amongst users to provide junk tags eg. abc, xyz should not work. There could be a website or public forum where users decide upon a global strategy to provide some given tags irrespective of the video they are watching. The scoring function should nullify such cases.

We only allow those tags to be specified which are present as a concept in *Wikipedia*. This ensures that junk would not be a part of tag set for any video. The factor of $\frac{tf\_idf}{max_t^v(tf\_idf)}$ because of division by maximum tf_idf ensures that after sufficient number of good tags have been collected for a given video the effect of junk tags and gaming the system is negligible.

But *this scoring function has certain inherent problems.* These are -

1. In calculating term frequency, $tf = \frac{f_v^t}{c_v}$, division by count of distinct tags is not necessary since this gets canceled out in the final score calculation

2. Presence of $\frac{tfIdf}{max_v(tfIdf)}$ factor is just to get score in range (0, 1). This demands extra information to be stored

3. In factor $\frac{tfIdf}{max_v(tfIdf)} = \frac{tf_1 * idf_1}{tf_2 * idf_2}$, let subscript 2 corresponds to max value tf_idf. Let tag 2 denote a relevant tag and tag 1 a unrelated tag. Now idf_2 can be reduced by tagging other videos with tag 2, thus increasing N_t, which would cause increase in score for some unrelated tag 1, which is not used for other videos, thus having higher idf_2. This is not desirable.

We try to improve upon the scoring function while preserving the desired properties.

2.4 Improvement of Scoring Function

We improve upon some of the components of the scoring function -

1. Term frequency is changed as $tf = \frac{f_v^t}{f_v}$. Division by maximum frequent tag ensures that tf of an unrelated tag would decrease as we have sufficient number of related tags.

2. Instead of normalization by division using $max_t^v(tf\_idf)$, we use division by $1 + logN$

3. Above tf and normalization ensure TF-IDF scores between 0 and 1 along with property that unrelated tag scores decreases with increase in sufficient number of related tags.

The new term frequency would be -

$$tf = \frac{f_v^t}{max_v(f_v^t)} \tag{2.6}$$

where max_v takes maximum value of frequency for the video v.
The new normalization term would be -

$$idf_{max} = 1 + \log N \tag{2.7}$$

Other values being same from the old scoring, the new scoring function becomes -

$$score = s_{watson} * w_{watson} * \frac{idf}{idf_{max}} * c_1 + (1 - w_{watson}) * \frac{tf\_Idf}{idf_{max}} * c_2 + c_3 \tag{2.8}$$

This new scoring function overcomes the problems we specified in the previous section 2.3.

1. There is no term which is cancelled out in this function and hence no unnecessary computation.

2. The normalization factor is a constant and cannot be influenced.

3. The new scoring function has desired properties from the previous scoring function and hence is good.

We next demonstrate the effect of both the functions over small data-set.

2.5 Experiments and Results

In the experimental setup we have users $S1$ to $S9$. Users go on to tag two videos with two tags - one relevant and other irrelevant. In one case the score is given by old scoring and in other case using the new scoring.

Old Scoring : After user **S4** completes tagging 9 different videos where tagged *Iron Man*, then user **S5** onward proceeded to re-tag this video.

Tag	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**
Iron Man	80	100	100	100	-	-	100	100	-
Metal	0	10	-	-	40	90	-	-	100

Table 2.2: Variation of Tag Score for Video - Will you be iron man?

New Scoring : After user **S4**, 9 different videos where tagged *Food*, then users **S5** onward proceeded to re-tag this video.

Tag	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**
Food	70	75	90	100	-	-	95	95	-
Shoe	0	0	-	-	20	60	-	-	75

Table 2.3: Tag Score for Video - Science of sweetness

'-' denotes this tag was not given
Observations *which reflect the improvements of new scoring function* -

1. The old scoring does not reflect clearly idf factor affecting the score. Even when 9 more videos where tagged with *Iron Man* there is no change in its score in user **S7** in Table 2.2 . But we can see in Table 2.3 there is dip in score of tag *Food*. Thus the new scoring function showing better sensitivity.

2. Old scoring allows unrelated tags have spiked increase in score which can be observed in the higher rate of increase in score for tag *Metal* with the number of tagging. The new scoring allows a gradual controlled increase and thus discourages users to given unrelated tags just to obtain higher scores.

All these establish that the new scoring function has the desirable properties which we stated in section 2.3. *The scoring function induces a ranking on the tags for a video.* Picking the tags after putting some threshold score would achieve video classification.

Chapter 3

MetaLearner

3.1 MetaLearners

We have seen in the previous chapter how TF-IDF scoring is used to combine the tags users give in order to achieve the video classification. But it has certain shortcomings.

1. There is no learning element involved. We just take the majority voting and account of specificity of the tags in order to rank the tags.
2. The TF-IDF model would only be able to suggest tags from the set of tags which people have suggested for a given video.
3. No correlation between tags or domain knowledge is utilized in scoring the tags.

We would like to now introduce some learning elements. We look for meta learning in this case because we do not have knowledge about the features the users looked into in order to give a particular tag set. We only have access to tag sets. Thus meta learners are perfect models for our problem.

We remodel our problem, in order to be able to use meta learning algorithms. So we assume following -

1. Each user is a base learner. A user has undergone real life training in inferring relevance of tags and items, here videos. Based on his experience he gives tag set for a given video. We treat it as output of his base learner.
2. User gives tags from a fixed set of global tags. This keeps the tag set free of concepts which are not known.
3. Classifiers based on images and metadata from videos are another base learner along with users.

Keeping these assumptions in mind we explore most popular metalearners and select the one which best suit to our problem.

3.2 Ensemble based Learners

As we see our problem is closely related to multi label multi class classification, Boosting which is an ensemble based learner is a popular choice because of its theoretical performance guarantees and strong experimental results. Ensemble learners are based on utilizing multiple learning algorithms to consolidate their results. Our problem fits into this as we assumed users to be base learners.

Most of the boosting algorithms are adaptations of *AdaBoost*[6] algorithm. So we analyze adaBoost to see possibility of using it in our setting.

AdaBoost[7] generates a sequence of base models $h_1, h_2, ..., h_M$ using weighted training sets (weighted by $D_1, D_2..., D_M$) such that the training examples misclassified by model h_{m-1} are given half the total weight when generating model h_m and correctly classified examples are given the remaining half of the weight. One of the major steps in this algorithm is calculation of weighted error for a base classifier, h_m, as -

$$\epsilon_m = \sum_{i:h_m(x_i)\neq y_i} D_m(i) \tag{3.1}$$

As we can see we need supervised data for training in this algorithm. Generating supervised data to train the model would require manual effort, since in our case we have crowd sourced data of video with tagset. Hence we abandon this idea of using adaboost based metalearners.

We would ideally want learners which focus of combination of model outcomes with maximum agreement amongst the learners. This brings us to consensus based learners which we explore in our next section.

3.3 Consensus based Learning

Combination of multiple models helps to overcome effects of poor data quality and also makes the final combined model more robust. In certain situation we may only have access to the predictions from the raw data and not the raw data itself. Consensus based prediction combination algorithms are effective for these situations.

We adapt the MultiLabel Consensus Maximization for ranking MLCM-r from [1] to achieve consensus amongst different users for the tags of a given video. We shall call it MLCM. As per authors of [1] "MLCM-r consolidates the predictions of base models via maximizing model consensus and exploits correlations using random walk in the label space." Let us see MLCM-r :

Table 3.1: Notations for MLCM-r

Symbol	Meaning
m	Number of multilabel classifiers
n	Number of Instances
l	Number of labels
$\boldsymbol{x}$	An instance
$\|.\|$	Frobenius Norm
A	$a_{i,j}$ is the prediction of label (j mod l) on $\boldsymbol{x}_i$ by $\lfloor j/l \rfloor$
B	Label node class distribution
U	$u_{i,l}$ is the probability that label l is relevant to $\boldsymbol{x}_i$
Q	$q_{i,l}$ is the probability of seeing label l given label j

We have predictions of m classifiers for n instances, with each prediction being a l length tuple consisting of 0's and 1's, with 1 at position i meaning that the ith label is relevant to this instance. This information of prediction is encoded in matrix A, which is n by v ($v = m \times l$), where $(i, (k-1) \times l + j)$-th entry is 1 if the k-th model predicts that the j-th label is relevant to the i-th instance, otherwise that entry is set to 0.

We construct a bipartite graph with A as the connection matrix. The graph has n instance nodes and $v = m \times l$ group nodes. A group node represents a label. The group nodes are annotated by letter g and the instance nodes by letter x. An instance node can be connected to more than one group nodes from a single classifier, representing the multilabel predictions of that classifier. A dashed rectangle surrounds group nodes which belong to the same classifier. There is another layer of nodes called label nodes which help represent the label the group node stands for. An example graph has been shown in Figure 3.1 for 2 instances, 3 labels and 2 classifiers.

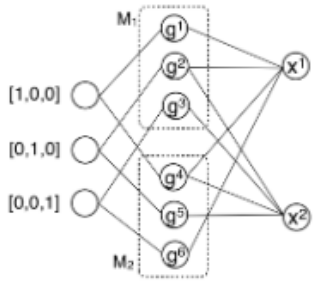

Figure 3.1: Bipartite Graph for MLCM-r[1]

Each node has a probability distribution associated with it.

1. Each instance node i is associated with probability distribution $\boldsymbol{u}_i$, where $u_{i,l}$ stands for probability of relevance of l-th label on i-th instance. Matrix $U = [u_1^{'}, ..., u_n^{'}]$ represents this information.

2. Each group node j is associated with probability distribution $\boldsymbol{q}_j$, where $q_{j,l}$ stands for probability of seeing label l given j-th label. This relates to how related two labels are in opinion the the classifier to which the group node j belongs. Matrix $Q = [q_1^{'}, ..., q_2^{'}]$ encodes this information.

3. Each group node j representing label l is also connected to label node which has probability distribution $\boldsymbol{b}_j$, where l-th entry is 1 and others 0. Matrix $B = [b_1^{'}, ..., b_v^{'}]$ encodes this information.

MLCM-r maximizes model consensus by solving the following optimization problem :

$$\min_{U,Q} \sum_{i=1}^{n} \sum_{j=1}^{v} a_{ij} \|\boldsymbol{u}_i - \boldsymbol{q}_j\|^2 + \alpha \sum_{j=1}^{v} \|\boldsymbol{q}_j - \boldsymbol{b}_j\|^2 \tag{3.2}$$

s.t.

$$u_{ik} \geq 0, \sum_{k=1}^{l} u_{ik} = 1, i = 1, ..., n \tag{3.3}$$

$$q_{jk} \geq 0, \sum_{k=1}^{l} q_{jk} = 1, j = 1, ..., v \tag{3.4}$$

In this the first term ensures that if an object $\boldsymbol{x}_i$ is linked to group $\boldsymbol{g}_j$, then their probability distribution should be similar since both of them in some sense represent probability of seeing label while being at that node. The second term ensures that the probability distribution of the groups does not deviate much from its initial distribution. α is the factor of how much we would like to penalize such a constraint violation.

The solution[2] is obtained by block co-ordinate descent :

$$\boldsymbol{q}_j^t = \frac{\sum_{i=1}^{n} a_{ij}\boldsymbol{u}_i^{t-1} + \alpha \boldsymbol{b}_j}{\sum_{i=1}^{n} a_{ij} + \alpha} \tag{3.5}$$

$$\boldsymbol{q}_j^t = \frac{\sum_{j=1}^{v} a_{ij}\boldsymbol{q}_j^t}{\sum_{j=1}^{v} a_{ij}} \tag{3.6}$$

Upon convergence, the final probability distributions are given in the rows of U. Most likely label can be decided from this.

We adapt this setting for our problem of classifying video. Our analogy consists of

1. A video is considered an instance $\boldsymbol{x}$
2. Each user is considered as multilabel classifier, since the users watch the videos and give the tags
3. Tags are correspondence of labels. We had earlier limited the tags from a fixed set of *Wikipedia* concepts which helps here.

For cases when a person does not tag a video, we assume that he does not associate it with any of the labels, and fill the connection matrix with entry of 0. After obtaining matrix U, we keep a threshold to obtain set of labels (tags) which are relevant to the given video.

Next we experiment with this algorithm on the data collected using the *tagVideo* game and analyze the results.

3.4 Experiments and Results

The experimental setting consists of *tagVideo* game where users provide tag set for videos. We use the same analogy as described in previous section 3.3. We also use *Kendall rank correlation coefficient* τ to measure the correlation between the rankings generated using TF-IDF scoring and MLCM-r scoring. The kendall's tau measures rank correlation: the similarity of the orderings of the data when ranked by each of the above method. This allows us to have a sense of how different MLCM-r becomes while it maximizes consensus.

We present the results in two parts each bringing out certain facet of MLCM-r.

CASE 1

Table 3.2: Result Table 1

Video Id	Tag-Score by TF-IDF	Tag-Score by MLCM-r	Correlation τ
guh7i7tHeZk	• ScienCe,1	• ScienCe,0.49 • Technology,0.26 • Play & Win,0.26	0.67
mtg9p6A6xnY	• Element 13,0.54 • Periodic Table,0.46	• Element 13,0.41 • Periodic Table,0.36 • Chemistry,0.23	1.0
w2Qk-jz_tWc	• Animals,1	• Animals,0.38 • Irrigation,0.15 • Crops,0.15 • Food,0.15 • Farming,0.15	0.4

We obtain following observation from Table 3.2 :

1. TF-IDF is only capable of ordering tags which where actually assigned to the video by the users.

2. MLCM-r also provides new tags which are related to original tags. Thus MLCM-r shows its capacity to capture label correlations. Correlations lead to appearance of tags like *Chemistry* with *Periodic Table*, even when no user had specified *Chemistry* as a tag.

CASE 2

Table 3.3: Result Table 2

Video Id	Tag-Score by TF-IDF	Tag-Score by MLCM-r	Correlation τ
649iUqrOKuE	• Big data,1	• Big data,0.28 • CERN,0.21 • Data storage,0.15 • Cloud Computing,0.15 • *Tag (graffiti)*,0.14 • *Tag (HTML)*,0.07	0.33

Table 3.3 gives us following insight :

1. MLCM-r not only brings in additional related tags, but it also brings in unrelated tags (*italicized in table 3.3*) which are profusely given with these related tags in some other videos.

2. The above limitation causes MLCM-r to provide a lot more tags than the optimal amount

We obtain average correlation of $\tau = 0.26$, which tells that it positively agrees with TF-IDF rank list.These results demonstrates capability of MLCM-r in capturing the label correlations, which help to bring in new relevant tags for the videos, while still maintaining good agreement with majority voting rank list.

Chapter 4

Conclusion and Summary

4.1 Summary

In this work, we have looked into how to utilize crowd sourcing in order to achieve video classification, with main objective of helping the farming community. We saw *TagVideo* game as an interface to collect data, and serving as a fun and learning element to players. We looked at the scoring mechanism and its improvements, so as to not allow users game the system.

We then introduced learning element into the game in form of metalearners. We analyzed popular metalearners and picked consensus learning for our problem. We then looked into working of MLCM-r and its improved tag prediction over TF-IDF. One application of MLCM-r apart from this problem could be in recommendation section, such as movie recommendation, where we need correlation between movies that users have watched.

It was a great learning experience to work on a problem with practical applicability. I learned a lot about metalearners, and had good experience of working in a group.

4.2 Future Work

In our work, we were able to use consensus learning algorithm, MLCM-r, to do video classification for crowd sourced data. But the method has limitations on which we would like to improve upon in our future work. The limitations with desirable improvements would be :

1. MLCM-r does allow for online updates, i.e. we need to compute the prediction from scratch in case some new edges are added. We would like to make the adapt the algorithm for online case to manage following :
    - Handle the adding of new edges due to users tagging videos which they had not tagged previously
    - New users being added into the system
    - Increase in number of tags. We would like our tag set to grow with time, so that we do not miss on new terms
    - New videos being added to the system
2. We have not utilized any domain knowledge. A prior knowledge graph containing the relations between tags is being currently built by fellow teammates, Depen and Shraddha. We would like to re-frame the optimization problem to be able to utilize the information.
3. We also would like to separate tags from concepts. This would give freedom to users to express tags in more specific free forms. Deepak in looking into working of mapping tags to concepts. We would like to change our model to utilize this transformation.
4. We would also like to improve the complexity of the algorithm, to handle situations when our video library grows big, and the system has lots of users.
5. We would also like to model the trust factor of users to attach weights to the tags he would be giving.

Consensus Learning for entire platform :
We would like to use consensus in all the platforms - Consensus amongst users to decide tags, consensus of tags predicted with the knowledge graph and consensus between user predicted tags and tags from algorithms based on textual and video, audio metadata.

These points summarize the future potential of the project.

Bibliography

[1] Sihong Xie, Xiangnan Kong, Jing Gao, Wei Fan, and Philip S. Yu. Multilabel consensus classification. In ICDM,2013.

[2] Jing Gao, Feng Liang, Wei Fan, Yizhou Sun, and Jiawei Han. Graph-based consensus maximization among multiple supervised and unsupervised models. In NIPS, 2009

[3] S. Cuendet, I. Medhi, K. Bali, and E. Cutrell. Videokheti: Making video content accessible to low-literate and novice users. ACM Conference on Human Factors in Computing Systems, April 2013

[4] Ramakrishna B. Bairi, Ankit Vani, Pooja Ahuja, Ganesh Ramakrishnan. Categorising videos using a personalised category catalogue. CoDS '15 Proceedings of the Second ACM IKDD Conference on Data Sciences

[5] C. Ramachandran, R. Malik, X. Jin, J. Gao, K. Nahrstedt, and J. Han. Videomule: A consensus learning approach to multi-label classification from noisy user-generated videos. In Proceedings of the 17th ACM International Conference on Multimedia, MM'09, pages 721-724, New York, NY, USA, 2009. ACM.

[6] Robert E. Schapire. Explaining Adaboost. pp 37-52, Book Title: Empirical Inference

[7] Oza, N.C. Online bagging and boosting. Published in: Systems, Man and Cybernetics, 2005 IEEE International Conference on (Volume:3)