Categorising videos using a personalised category catalogue

Ramakrishna B Bairi
IITB-Monash Research Academy
IIT Bombay
Mumbai, India
bairi@cse.iitb.ac.in

Ankit Vani
Research Intern
Department of CSE
IIT Bombay
Mumbai, India
ankit@nevitus.com

Pooja Ahuja
Research Intern
Department of CSE
IIT Bombay
Mumbai, India
p@ahujapooja.com

Ganesh Ramakrishnan
Department of CSE
IIT Bombay
Mumbai, India
ganesh@cse.iitb.ac.in

ABSTRACT

Video is an extremely effective way of reaching farmers with the latest agricultural technology and stories of other farmers. With a well-organised multifaceted video library, we can provide the farmers with services such as easy navigation, search and recommendations of videos as per their needs. Since categories and tags assigned by video uploaders on YouTube are often prone to noise and are not uniform across a video collection, we propose a semi-automated system to achieve this desired video categorisation. We adopt active learning as our strategy to evolve a personalised category catalogue for agricultural videos.

We present a multi-label classification system, with a large global category catalogue, such as Wikipedia, as its initial label space. We narrow down on a domain-specific category catalogue, evolving our system by using associative Markov networks with the categories as nodes. Each node incorporates structural constructs and a binary SVM classifier as node features. We show that our categorisation for agricultural videos is less granular and more uniform as compared to YouTube tags, while staying sufficiently specific.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning—*concept learning, knowledge acquisition, parameter learning*; I.7.m [**Document and Text Processing**]: Miscellaneous

General Terms

Algorithms, Theory

CODS '15 March 18 - 21, 2015, Bangalore, India

http://dx.doi.org/10.1145/2732587.2732594

Keywords

Active learning, large scale classification, multi-label classification, personalisation, topic identification, video categorisation

1. INTRODUCTION

1.1 Motivation

Videos can be one of the most convenient sources of information for farmers to stay up to date with the latest techniques, possibilities and success stories from others all around the world. A video library can easily be accessed and utilised by farmer groups using specialised mobile applications, since smartphones, which were once an oddity in rural areas, have now become commonplace. If a farmer has a problem in the field, a video relevant to the problem can be recommended to him/her. We aim to facilitate such a video library, which can grow as more videos become available from various sources.

We have manually collected over two thousand URLs from YouTube for videos related to agriculture. These videos belong to various categories that range from broad farming concepts such as rainwater harvesting to specific videos such as plantation techniques for sugarcane. Using the facets assigned by us for the videos, a farmer can search or be recommended videos as per his/her need. Apart from broad-level categories, these facets would also include aspects such as the language of the video, the crop(s) discussed, the machinery involved, and the like. This motivates the need for an evolving label space or a *personalised category catalogue*, along with a corresponding categorisation mechanism. We aim to categorise agricultural videos into fine-grained facets, such that they can be organised for easy search and navigation from a mobile application.

The video repositories currently in place, such as *YouTube*, provide a general-purpose solution for the task of video categorisation and searching. This means that a closed domain, such as that of agriculture, falls into a small cluster, or is at best a union of a handful of clusters of all of YouTube videos.

Every video on YouTube can be assigned exactly one *category*, which is chosen by the uploader from a fixed list of high-level domains like Education, Comedy, Music, Travel, and such others. Such a high-level categorisation is not adequate when dealing with a closed user group with more focused needs.

Along with a category, uploaders on YouTube also specify a list of *tags* for their videos. Although tags are usually more specific to the video, we observed that they are often too granular, and most of them do not re-occur in other videos, as seen in Figure 1. Additionally, some videos may have very few or no tags, or the chosen tags may not suit our purpose for categorisation. Sometimes, uploaders also tend to add unrelated tags to their videos. As examples, the categories and tags for two videos are illustrated in Table 1.

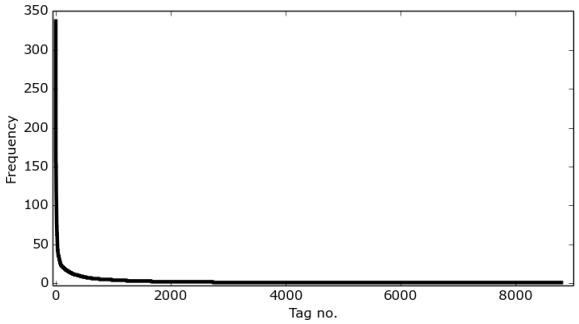

Figure 1: Distribution of YouTube tags arranged in decreasing order of frequency in agricultural videos in our collection.

Title:	Preserving Apple Varieties | What is Organic | Video | Veria Living
YouTube categ.:	Travel
YouTube tags:	Apple Varieties, apples, natural wellness, healthy lifestyle, natural, prevent, curious, curiosity, travel, discover, organic, farm, food, industry, dairy, adventureworld, country, Morocco, India, Asia, Europe, globe, vineyards, Italy, farming, eat, grow, extinct, fruits, vegetables, oil, oasis, methods, Himalayas, Organic Farming, What is Organic, video, What is Organic video, Organic, videos, Veria, Veria Living
Title:	Red Rome Apple Tree
YouTube categ.:	Howto
YouTube tags:	tyty nursery, ty ty nursery

Table 1: Two YouTube videos as examples, with their categories and tags.

Since both the categories and tags on YouTube are user-provided, they tend to have a lot of noise. From the video URLs that we collected for the agriculture domain, we calculated the information gain $IG(T, c)$ as the change in entropy of tags T on choosing a category c:

$$IG(T, c) = H(T) - H(T|c)$$

Category	**# of videos**	**# of tags**	**Gain %**
comedy	4	7	76.217%
animals	68	663	29.067%
people	183	1690	20.479%
travel	207	2210	15.792%
entertainment	69	745	26.925%
howto	409	4326	16.610%
sports	4	52	52.033%
autos	8	116	42.631%
tech	263	3377	18.906%
music	10	154	41.491%
news	192	3064	16.047%
nonprofit	160	1525	20.800%
education	511	5532	9.438%
film	29	298	33.993%
shows	10	89	52.163%
Avg. Gain	2127	23848	**17.066%**

Table 2: Information Gain from YouTube categories in agricultural videos in our collection.

where

$$H(T) = -\sum P(t_i) \log \big(P(t_i)\big)$$

$$H(T|c) = -\sum P(t_i|c) \log \big(P(t_i|c)\big)$$

The average information gain on tags by using YouTube categories is only 17.066%, as illustrated in Table 2. Ignoring tags that occur only once, this figure drops to 14.9%.

For reliable categorisation of videos, where each category for a video can be treated as a facet for navigation and search, we need to find a categorisation for videos at a granularity that is in between that of YouTube's categories and tags. Additionally, we aim to reduce the noise and the lack of uniformity that is present in YouTube's categorisation system.

We propose a system that is aimed at evolving a domain-specific category catalogue (such as for agriculture) from a large global category catalogue (GCC), assigning facets to videos from the domain-specific category catalogue, and providing a decent balance between granularity and cardinality of these facets. Categorisation systems that employ active learning can achieve a performance that is comparable to supervised classifiers, while using much less labelled data. This becomes even more important in multi-label classification because the user would need to label for all the possible categories for each instance to train otherwise. Thus, we propose a joint active learning technique over video and category spaces to train the model parameters of our system. This technique will solicit user feedback only for the most uncertain categories of the most uncertain videos.

1.2 Related Work

There have been attempts at fine-grained categorisation of videos such as [1] and [5]. However, they are often domain-specific and cannot fully be adapted to a different domain of videos. In contrast, the system we propose can be easily adapted to a different genre of videos.

A notable example of an attempt at presenting agricultural videos to farmers is *VideoKheti*[4]. However, in the case of VideoKheti, the main aim was not to categorise

videos, but to acquire data on the user experience and the usability of a multi-modal interaction system for searching and viewing of agricultural videos.

A large-scale video taxonomic classification scheme is presented in [10]. Here, a taxonomic structure of categories is deployed in the classifier training, considering both the video's content-based and text-based features. To compensate for the lack of labelled video data, classifiers trained on web-text documents are adapted to the video domain to leverage the availability of a large corpus of labelled text documents.

VideoMule[9] combines individual classification and clustering algorithms trained on textual metadata, audio and video through a heuristic consensus learning approach, thus assigning multiple semantic labels to the videos.

In this paper, we propose a solution that uses an approach similar to that of *EVO*[2, 3], a document classifier. The input for EVO is a huge global category catalogue (GCC), such as Wikipedia. The nodes of a category catalogue are the possible categories that may be assigned to a document being classified, each of which corresponds to a Wikipedia article or category. Each node has a description associated with it. The edges in a category catalogue are the associations between the Wikipedia articles/categories represented by the nodes. In our previous work[3], we showed that EVO outperforms the multi-label SVM in a warm-start setting for categorising documents. We also compared our joint active learning technique for documents with other active learning techniques in literature, and showed that we needed significantly less feedback to reach a particular F1 score, resulting in a reduced cognitive load on the user.

2. OVERVIEW

2.1 Video metadata

A video has some metadata associated with it: a title, description, category, and tags. We use a video's metadata as the features for performing classification in our system. The metadata is also used to select a subset of the GCC for the facet discovery process. This can be done using a *spotter*, which can perform topic detection on the metadata text with high recall to select categories from the GCC that may be relevant to a given video.

We designed a crawler to fetch the metadata from the URLs of agricultural videos that we collected from YouTube.

2.2 Evolving a domain-specific category catalogue

We use *Wikipedia* to build a global catalogue of categories $C = \{C_i\}_{i=1}^{i=f}$. The English Wikipedia has a large collection of articles numbering over 4.5 million. This makes it a good candidate to cover almost any terminology in English text. We receive videos in batches $V_1, V_2, \ldots$ and need to adopt the GCC to logically build a domain-specific category catalogue $C^{agri} \subseteq C$, and at the same time, evolve some models to classify all $v_i \in V_j$ into C^{agri}. Additionally, during production deployment, a *domain classifier* ensures membership of a category C_j in C^{agri}. Our system considers the following goals:

1. Learn a model to localise the GCC.
2. Build an evolving multi-label, multi-class video categorisation system to categorise videos v into C^{agri}.
3. Curate training data through active learning that facilitates localisation of the GCC.

v or v_i	Single input video
V or V_i	Single batch of input videos
C^{agri}	Agriculture-specific category catalogue
C_i	Category or facet in GCC represented by the i^{th} node in the Markov network (MN)
$\mathbf{x}_i$	Feature vector for i^{th} node (C_i) in the MN
$\mathbf{x}_{ij}$	Feature vector for the edge connecting i^{th} and j^{th} node in the MN
$\mathbf{x}$	Set of all node and edge feature vectors in the MN
y_i	Node label $\in \{0, 1\}$
$\mathbf{y}$	Set of all node labels in the MN
φ_i	Node potential of i^{th} node in the MN
ψ_{ij}	Edge potential of the edge connecting i^{th} and j^{th} node in the MN

Table 3: Important notations used in this paper.

The problem is to identify suitable categories $\{C_1, \ldots, C_T\}_v$ for a video v, that would serve as facets for the video library.

We can leverage the relationships between categories to build a robust model, in which the relationships serve two important roles: *i)* When we just start to categorise the videos, the model is not tuned to localise the categories. The only information available to the categorisation system is the category evidences. It is impractical to assume that perfect evidences will be available for every category. In such cases, the relationships between the categories can help propagate evidences across categories through their neighbours. *ii)* As part of learning model parameters, the system solicits user feedback on some of the suggested categories for a video. Based on the feedback, the category-specific models (SVM) are updated. Thus, the category relationships help in propagating the learning to the neighbours. This reduces the number of feedbacks needed to learn the model parameters.

Markov networks constitute one of the most intuitive options to account for the neighbourhood information in a category catalogue. For a given video, we model the space of GCC as a Markov network, in which the nodes represent the categories $C_i \in C$, and edges represent the relationships between the categories. As the system evolves, the nodes representing the categories that are not relevant to the agriculture domain become insignificant. Thus, we converge to a more personalized catalogue of categories to build a Markov network from. We assign a binary label $\big(0/1\big)$ for every node C_i in this Markov network. Label 1 indicates that the category C_i is valid for the video v, whereas, the label 0 indicates that C_i is invalid for the video. The collective assignment of labels for all the nodes in the Markov network produces the relevant categories or facets for the video v. As we see later in the paper, an optimal assignment of these labels can be achieved through MAP inference using Integer Linear Programming. Note that the Markov network built from all the categories in GCC can be extremely large, leading to intractable inference. However, most of the nodes can be pruned affront, making the Markov network inference practical.

We present the steps of our system's overall facet discovery

Algorithm 1 Batch Facet Discovery Process

1: **Input:** Global Category Catalogue $(C_1, \ldots, C_f)$, Video batch V_j, Previously-learnt Model parameters θ^t
2: **Output:** Categories relevant to each video $v_i \in V_j$, Updated model parameters θ^{t+1}
3: **for all** $v_i \in \{V_j\}$ **do**
4: Retrieve categories from GCC and prune uninteresting categories for v_i
5: Build an associative Markov network (AMN) of categories from the GCC
6: Compute node potentials φ_i and edge potentials ψ_{ij} using the model parameters θ^t and the structural constructs for the AMN
7: Perform 0/1 inference on the above AMN. Assign all categories labelled as 1 as the facets for video v_i
8: **end for**
9: Select the most uncertain videos and categories for feedback, and do joint Active Learning in video and category space
10: Solicit user feedback F for the selected videos and categories and update model parameters, using learner $\mathcal{L}$

$$\theta^{t+1} = \mathcal{L}\left(\theta^t, F\right)$$

process in Algorithm 1.

3. VIDEO CATEGORISATION USING AMN

3.1 Markov Random Field

A Markov Random Field (MRF), also known as Markov network (MN) is an undirected graph with a set of cliques $\mathbb{C}$, and a *clique potential* c, which is a non-negative function associated with each $c \in \mathbb{C}$. In the context of classification, we consider conditional MRFs that define the distribution

$$P(\mathbf{y}|\mathbf{x}) = \frac{\prod_{c\in\mathbb{C}} \phi_c\left(\mathbf{x}_c, \mathbf{y}_c\right)}{\sum_{\mathbf{y}'} \prod_{c\in\mathbb{C}} \phi_c\left(\mathbf{x}_c, \mathbf{y'}_c\right)} \tag{1}$$

where $\mathbf{x}_c$ and $\mathbf{y}_c$ are the features and labels of nodes in the clique c. Here, the potential ϕ_c is a mapping from features and labels to a positive value. The higher the value, the more likely it is that the labels $\mathbf{y}_c$ are correct for the features $\mathbf{x}_c$. The denominator in equation 1 is called the partition function, usually denoted by Z, and is essentially a sum over all possible labellings.

3.2 Associative Markov Network

To simplify the problem, the size of the cliques is usually restricted to be either one or two. This results in a pairwise MRF, where only node and edge potentials are considered. For a pairwise MRF with the set of edges E, equation 1 simplifies to

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z}\prod \varphi\left(\mathbf{x}_i, y_i\right) \prod \psi\left(\mathbf{x}_{ij}, y_i, y_j\right) \tag{2}$$

Here the node features $\mathbf{x}_i$ are computed by considering the node evidence and the input video v. Again, Z denotes the partition function given by $Z = \sum_{\mathbf{y}'} \prod \varphi\left(\mathbf{x}_i, y_i'\right) \prod \psi\left(\mathbf{x}_{ij}, y_i', y_j'\right)$. Note that in equation 2, there is a distinction between node features $\mathbf{x}_i \in \mathbb{R}^{v_n}$ and edge features $\mathbf{x}_{ij} \in \mathbb{R}^{v_e}$.

A simple way to define the potentials φ and ψ is the log-linear model. In this model, a weight vector w_k is introduced for each class label $k = 1, \ldots, K$. The node potential φ is then defined as $log\varphi\left(\mathbf{x}_i, y_i\right) = \mathbf{w}_n^k \cdot \mathbf{x}_i$, where $k = y_i$. Accordingly, the edge potentials are defined as $log\psi\left(\mathbf{x}_{ij}, y_i, y_j\right) = \mathbf{w}_e^{k,l} \cdot \mathbf{x}_{ij}$, where $k = y_i$ and $l = y_j$. Note that there are different weight vectors $\mathbf{w}_n^k \in \mathbb{R}^{v_n}$ and $\mathbf{w}_e^{k,l} \in \mathbb{R}^{v_e}$ for the nodes and edges.

We can express the potentials as

$$log\varphi\left(\mathbf{x}_i, y_i\right) = \sum_{k=1}^{K} \left(\mathbf{w}_n^k \cdot \mathbf{x}_i\right) y_i^k \tag{3}$$

$$log\psi\left(\mathbf{x}_{ij}, y_i, y_j\right) = \sum_{k=1}^{K} \left(\mathbf{w}_e^{k,l} \cdot \mathbf{x}_{ij}\right) y_i^k y_j^l \tag{4}$$

where y_i^k is an indicator variable that is 1 if the i^{th} node has the label k, and 0 otherwise.

To bring in the notion of association, we introduce the constraints $\mathbf{w}_e^{k,l} = 0$ if $k \neq l$ and $\mathbf{w}_e^{k,k} \geq 0$. This results in $\psi\left(\mathbf{x}_{ij}, k, l\right) = 1$ if $k \neq l$ and $\psi\left(\mathbf{x}_{ij}, k, k\right) \geq 1$. The idea here is that edges between nodes with different labels should be penalised over edges between equally labelled nodes.

The objective is to maximise $P\left(\mathbf{y}|\mathbf{x}\right)$, which is equivalent to maximising $log\, P\left(\mathbf{y}|\mathbf{x}\right)$. By substituting 3 and 4, the objective becomes

$$\max_{\mathbf{y}} \sum_{i=1}^{N}\sum_{k=1}^{K} \left(\mathbf{w}_n^k \cdot \mathbf{x}_i\right) y_i^k + \sum_{(ij)\in E}\sum_{k=1}^{K} \left(\mathbf{w}_e^{k,k} \cdot \mathbf{x}_{ij}\right) y_i^k y_j^k - \log Z\left(x\right) \tag{5}$$

Note that the partition function Z only depends on $\mathbf{w}$ and $\mathbf{x}$, but not on the labels $\mathbf{y}$, which need not be computed while solving the optimisation problem mentioned above.

3.3 Feature Engineering

The node $(\mathbf{x}_i)$ and edge $(\mathbf{x}_{ij})$ feature values are the primary factors in determining the node labels. The localisation of the GCC is also controlled by the node features. We describe some of the features that we found useful in our setting.

3.3.1 Node features

Node features determine the relevance of a category to the input video v. We divide the node features $\mathbf{x}_i$ into two categories: *i)* Static node features $\mathbf{x}_i^s$ and *ii)* Dynamic node features SVM_i^0 and SVM_i^1. Thus, the node feature vector is $\mathbf{x}_i = \left[\mathbf{x}_i^s; SVM_i^0; SVM_i^1\right]$.

As the system evolves, the values of static features do not change, whereas, the values of dynamic features change. Static features aid in capturing the structural similarity through the combination of different kernels. In the absence of dynamic features, static features help in determining the similarity of a node to the input video. This situation arises when a category is being considered for categorisation for the first time. Dynamic features aid in the localisation of GCC. These feature values are computed from the category-specific parameters that are continuously updated based on active learning and user feedback. When the categorisation system is initially deployed, there are no localisation parameters. Hence, the dynamic features do not contribute in deciding the relevance of a category. As the system evolves, we

expect more importance to be associated with the dynamic features. This happens due to learning of updated feature weights from AMN, SVM retraining and γ_i re-estimation.

Bag of Words kernel (Static feature).

In this most widely used feature representation, both the category evidence and input video's metadata is represented as a TF-IDF vector, and cosine similarity is computed between these vectors. However, we use *decayed term frequency*, where the repeated occurrences of the same word in a video's metadata do not contribute equally to term frequency. The effective term frequency is $\sum_{i=1}^{K} e^{1-i}$, where K is the number of times the term occurs in the video metadata. This formulation is based on the observation that, once an instance of a term is observed in a video's metadata, it is very likely that more instances of the term will be observed in the same metadata. Hence, multiple occurrences of the same term in a video's metadata will not artificially boost the cosine similarity.

Structural Constructs (Static features).

The structural constructs in the GCC can help determine the similarity of a category to an input video. For example, the degree of overlap between hyperlink anchor text in the category evidence and the input video metadata can determine the similarity strength. We use the notation $S_i^{C_j}$ to denote the i^{th} structural construct associated with category C_j. Let $S^{C_j} = \left\{ S_1^{C_j}, S_2^{C_j}, \ldots, S_{|S^{C_j}|}^{C_j} \right\}$ refer to the structural constructs associated with category C_j in the GCC. We denote $K\left(S_j^{C_i}, v\right) \rightarrow \mathbb{R}$ as the function that computes the similarity between the input video v and the j^{th} structural construct of the i^{th} category, returning a real value. Many such kernels can be defined based on the different kinds of structural constructs available in the catalogue.

Category-specific Classifiers (Dynamic feature).

We incrementally train an SVM for each category in C^{agri}, and employ its decision function in two node features. Although we have chosen SVM, it is possible to consider the use of other machine learning models such as decision trees, logistic regression, and the like. Our choice of SVM was based on the fact that SVMs are known to yield high accuracies.

3.3.2 Edge features

Edges between categories in a Markov network represent some kind of association between them. An Edge feature vector ($\mathbf{x}_{ij}$) contains feature values that encourage the categories C_i and C_j connected by the edge to have the same label if there is a strong relationship between them. The strength of relationship is discovered through combinations of multiple kernels. Let $K_1(C_i, C_j), \cdots, K_M(C_i, C_j)$ be M kernels that measure the similarity between C_i and C_j. Example kernels include Bag-of-Words kernel, Bi-gram kernel, Trigram kernel, Relational kernel, and the like. Further, we assume (without loss of generality) that these kernels return normalised values (between 0.0 and 1.0). We define the feature vector $\mathbf{x}_{ij}$ to have M features that signal $y_i = 1$, $y_j = 1$ and M features that signal $y_i = 0$, $y_j = 0$. The feature vector $\mathbf{x}_{ij}$ is defined as follows:

$\forall 1 \leq m \leq M$,

$$\mathbf{x}_{ij}[m] = K_m(C_i, C_j) \times \left(\log \varphi(\mathbf{x}_i, 1) + \log \varphi(\mathbf{x}_j, 1)\right)$$

$$\mathbf{x}_{ij}[M+m] = K_m(C_i, C_j) \times \left(\log \varphi(\mathbf{x}_i, 0) + \log \varphi(\mathbf{x}_j, 0)\right)$$

Note that $\log \varphi(\mathbf{x}_i, 1)$ is the node potential of C_i when it is labelled 1 and $\log \varphi(\mathbf{x}_i, 0)$ is the node potential when it is labelled 0. Essentially, when the similarity between nodes C_i and C_j is high, the edge features collectively favour the label 1 on both the nodes C_i and C_j, or the label 0.

3.3.3 Learning feature weights

Learning the feature weights is based on Max-Margin training as detailed in [11]. The quadratic program (QP) is of the form:

$$\underset{\mathbf{w},c}{\text{argmin}} \ \frac{1}{2}\|\mathbf{w}\|^2 + c\xi$$
$$s.t.\ \mathbf{w}\mathbf{X}\widehat{\mathbf{y}} + \xi \geq \max_{\mathbf{y}} \mathbf{w}\mathbf{X}\mathbf{y} + \left(|N| - \mathbf{y}.\widehat{\mathbf{y}}_{\mathbf{n}}\right);\ \ w_e \geq 0$$

3.3.4 Handling unbalanced classes

The training data is often highly skewed. Usually, a video will have only a few relevant categories. Hence, most of the category nodes in AMN will have the label 0, and only a few will have the label 1. In order to handle this skewness in the training data, we explored two strategies: *i)* penalty variation, and *ii)* node boosting.

Penalty Variation.

We introduce two hyperparameters C_p and C_n that award different penalties for a label 1 mismatch and a label 0 mismatch in the max-margin training objective:

$$\min \frac{1}{2}\|\mathbf{w}\|^2 + c\xi$$
$$s.t.\ \mathbf{w}\mathbf{X}\widehat{\mathbf{y}} + \xi \geq \max_{\mathbf{y}} \mathbf{w}\mathbf{X}\mathbf{y} + C_p\left(N^1 - \mathbf{y}^{\mathbf{1}}.\widehat{\mathbf{y}}_{\mathbf{n}}^{\mathbf{1}}\right)$$
$$+C_n\left(N^0 - \mathbf{y}^{\mathbf{0}}.\widehat{\mathbf{y}}_{\mathbf{n}}^{\mathbf{0}}\right);\ \ w_e \geq 0$$

where N^1 and N^0 are total number of 1's and 0's in the training examples, and $\mathbf{y}^{\mathbf{1}}.\widehat{\mathbf{y}}_{\mathbf{n}}^{\mathbf{1}}$ are the total number of disagreements between actual label 1's and inferred label 1's. Similarly, $\mathbf{y}^{\mathbf{0}}.\widehat{\mathbf{y}}_{\mathbf{n}}^{\mathbf{0}}$ are the disagreements between actual label 0's and inferred label 0's. Empirically, we have found that training with this technique is highly sensitive to the hyperparameters C_p and C_n. Setting $C_n = 1$ and C_p = (# nodes with label 0) / (# nodes with label 1) generally yields better results.

Node Boosting.

In this technique, we replicate all the nodes labelled 1 (and edges labelled $\{1,1\}$) multiple times in the training examples, so that total number of nodes labelled 1's and 0's are almost equal. We use this modified AMN for training as well as for inference. Empirically, we have found this technique to be far more useful than penalty variation.

3.4 Inference

The problem of inference is to determine the most relevant categories for a new input video. That is, we need to

select a subset of nodes from the category catalogue, whose corresponding categories have the highest probability of being relevant to the input video. To model this selection, we attach a binary label $\{0, 1\}$ to a node. A node C_i with label 1 is considered to be a valid category for the input video, and invalid if its label is 0. The node and edge potentials are calculated as in (3) and (4) with $K = 2$.

Correctly determining the categories for an input video is equivalent to solving the optimisation problem (from [11] and [3]):

$$\max_{\mathbf{y}} \sum_{i=1}^{N} \sum_{k=0}^{1} \left(w_n^k \cdot x_i\right) y_i^k + \sum_{(ij)\in E} \sum_{k=0}^{1} \left(w_e^k \cdot x_{ij}\right) y_{ij}^k \quad (6)$$

$$s.t. \quad y_i^k \geq 0, \quad \forall i, k \in \{0, 1\}\,;$$

$$\textstyle\sum_{k=0}^{1} y_i^k = 1, \quad \forall i$$

$$y_{ij}^k \leq y_i^k, \quad y_{ij}^k \leq y_j^k, \quad \forall ij \in E, k \in \{0, 1\}$$

$$y_i^0 \quad = 1 \quad \forall i \text{ with Hard Constraints}$$

The variables y_{ij}^k represent the labels of two nodes connected by an edge. The last two inequality conditions are a linearisation of the constraint $y_{ij}^k = y_i^k \wedge y_j^k$. Hard Constraints are explained in section 4.1.5.

3.5 Data reduction

With an extremely large AMN, it becomes prohibitive to compute node/edge features and run the MAP inference. A video generally spans a small number of categories. Therefore, it is unnecessary to instantiate a node in the MN for all the categories in GCC (to only later discover that most of their labels have been inferred to be 0's). We can affront discard most of the unnecessary categories for the input video from the GCC and retain only a sizeable number of categories to construct the AMN. We refer to this phase as the *candidate selection* phase, and further identify two stages within this phase: *i) Keyphrasing:* Identifying important words and phrases from the input video, and *ii) Candidate detection:* Selecting a set of categories from the GCC that relate to these phrases/words.

3.5.1 Candidate detection

Key phrases detected in the previous step are used to lookup an index of category titles and evidences to retrieve candidate categories. At this stage, we do not disambiguate the retrieved categories. Irrelevant categories (with incorrect meaning with respect to the context of the input video) get low node feature values and get eliminated during AMN inferencing.

Candidate detection is performed in our system by a *spotter*. We chose *WikipediaMiner*[8] as a suitable candidate for this purpose, which can perform topic detection on the video metadata with good recall for selecting Wikipedia articles and categories that may be relevant to an input video.

4. LEARNING TO LOCALISE

As discussed above, the categorisation system suggests categories by virtue of solving the inference problem in (6). When the system is initially deployed, the categories that are suggested depend completely on the static features. It is quite possible for the system to suggest categories that may not be acceptable to the user. The following are some plausible reasons: *i)* the evidence for each category in the GCC is not exhaustive, *ii)* static node features are not discriminative across categories, leading to inter-category confusion, *iii)* the user may not be interested in certain (classes of) categories, such as those not belonging to the agriculture domain.

To match the user's expectations, we take feedback in the form of 'relevant', 'irrelevant', 'possibly relevant', and 'completely irrelevant' for the categories suggested by the system for a chosen video v and update the model accordingly. More precisely, we learn a per-category SVM using training data incrementally obtained during user feedback.

4.1 Active Learning

In the process of providing feedback for a video v, the user needs to mark every category suggested (by the system) for every video, as 'relevant', 'irrelevant', 'possibly relevant', or 'completely irrelevant'. This can produce a lot of cognitive load on the users. To reduce this cognitive load and to achieve a better learning rate, we seek feedback from the user on select categories for select videos. We incorporate information from this feedback for retraining the AMN and SVM model parameters. We adopt uncertainty sampling techniques to identify the most uncertain categories for a video as well as the most uncertain videos in a batch. In the following sections, we present our approach to simultaneous active learning in video and category spaces.

4.1.1 Deciding uncertain categories

We first define some basic concepts and then delve deeper into our techniques for active learning.

DEFINITION 1. *Uncertain Node: A category node C_i in an AMN is more uncertain than a node C_j if* $\left|P\left(y_i = 1\right) - P\left(y_i = 0\right)\right| < \left|P\left(y_j = 1\right) - P\left(y_j = 0\right)\right|$.

DEFINITION 2. *Influencing Node: A category node C_i in an AMN is an influencing node if there exists a node C_j in the neighbourhood of C_i, such that $y_j = k$ if $y_i = k$ for $k \in \{0, 1\}$.*

DEFINITION 3. *Inveigled Node: A category node C_i in an AMN is an inveigled node if there exists a node C_j in the neighbourhood of C_i, such that $y_i = k$ if $y_j = k$ for $k \in \{0, 1\}$.*

Notion of Label Flipping.

Let $Y = [y_1, \cdots, y_i, \cdots, y_n]$ be the labelling of all n nodes in the AMN that yields the solution to the inference problem in (6). Each $y_i = 0$ or 1. Consider flipping the label of i^{th} node to $\overline{y}_i$, where $\overline{y}_i = \begin{cases} 1 & \text{if } y_i = 0 \\ 0 & \text{if } y_i = 1 \end{cases}$.

Let $Y_i = \left[y_1', \cdots, y_i' = \overline{y}_i, \cdots, y_n'\right]$ be the labelling obtained by re-running the inference in equation (6) after this flip.

The quantity $\triangle Y_i = n - \sum_{i=1}^{n} \delta\left(y_i, y_i'\right)$, (where δ is the Kronecker delta function) indicates the number of label flips in Y_i with respect to the Y resulting from the flipping of the label of the i^{th} node.

If flipping of label y_i in Y results in flipping of y_j in Y_i, then there exists a path in the MN from node C_i to C_j on which every node's label is flipped in Y_i. That is, if C_k is a node on such a path, then $y_k' = \overline{y}_k \ \forall k$.

Algorithm 2 Flip-Test Score Generation

function FLIP_TEST_SCORER($C, \mathbf{y}, \mathbf{w}$)
  Input: AMN of categories $C = (C_1, \ldots, C_n)$, inferred labels $\mathbf{y} = (y_i, \ldots, y_n)$, feature weights $\mathbf{w} = \left(w_n^0, w_n^1, w_e^0, w_e^1\right)$
  Output: Flip-Test score for every node
  Initialise $R[C_i] = 0 \; \forall i = 1, \ldots, n$
  for all $C_i \in C$ **do**
    Initialise queue Q with C_i
    Initialise $V = \{\emptyset\}$
    $R[C_i] = 1$
    while Q **not** *Empty* **do**
      $T = Q.pop()$ ▷ Pick next item from queue
      Flip the label of T
      for all $N_i \in Nbr(T)$ **and** $N_i \notin V$ **do**
        Let z_i = label of N_i and $\overline{z}_i = \sim z_i$, flipped label of N_i
        $$\phi = w_n^0.x_i.\delta(z_i, 0) + w_n^1.x_i.\delta(z_i, 1) + \sum_{j \in Nbr(N_i)} \left(w_e^0.x_{ij}.\delta(z_i, 0).\delta(z_j, 0) + w_e^1.x_{ij}.\delta(z_i, 1).\delta(z_j, 1) \right)$$
        $$\overline{\phi} = w_n^0.x_i.\delta(\overline{z}_i, 0) + w_n^1.x_i.\delta(\overline{z}_i, 1) + \sum_{j \in Nbr(N_i)} \left(w_e^0.x_{ij}.\delta(\overline{z}_i, 0).\delta(z_j, 0) + w_e^1.x_{ij}.\delta(\overline{z}_i, 1).\delta(z_j, 1) \right)$$
        if $\overline{\phi} > \phi$ **then**
          Flip the label of N_i
          $R[C_i] = R[C_i] + 1$
          $Q.push(N_i)$ ▷ Add N_i to queue
        end if
      end for
      Add T to V
    end while
  end for
  return R
end function

Computing Y_i for every node is computationally expensive. We would need to run MAP inference once for every node after fixing its label. Thus, we use an approximation algorithm. For each node, we flip its label and estimate if it results in flipping of any of its neighbour's labels. If so, we repeat the procedure for all the neighbours whose labels have been flipped. Algorithm 2 outlines this procedure. Empirically, we have found that the approximation error is less than 5%.

As shown in [2] and [3], there exists a feature space, and a hyperplane in that feature space passing through the origin, which separates the nodes with label 1 from the nodes with label 0.

PROPOSITION 1. *If a node C_i has higher $\triangle Y_i$ than a node C_j, then C_i is more influencing than C_j.*

PROPOSITION 2. *If a node C_i labelled 1 (or 0) in AMN has strongly associated neighbours that are labelled 0 (or 1), it is more likely to be an uncertain node than a node with neighbours that have identical labels.*

PROPOSITION 3. *A node with $y_i = 1$ and $log\varphi(x_i, 1) < log\varphi(x_i, 0)$ is more likely to be an inveigled node. So is a node with $y_i = 0$ and $log\varphi(x_i, 0) < log\varphi(x_i, 1)$.*

PROPOSITION 4. *A node with lower margin distance*

$$m_i = \left| \left(w^1.\mathbf{x}_i + w^{11}. \sum_{j \in Nbr_1(x_i)} \mathbf{x}_{ij} \right) - \left(w^0 \mathbf{x}_i + w^{00}. \sum_{j \in Nbr_0(x_i)} \mathbf{x}_{ij} \right) \right|$$

is more uncertain.

We prepare a ranked list of nodes to seek feedback. Influencing and inveigled nodes are placed at the top of the list. If there are many such nodes, we order them by increasing margin distances m_i.

4.1.2 Deciding uncertain videos

In the previous section, we identified the most uncertain categories for each video to be categorised. We now make use of this to identify the most uncertain videos.

The association between videos and categories can be represented as a bipartite graph, in which each video is connected to its L most uncertain categories.

Our classifier's approach is to select a subset of videos that results in the maximum coverage of the most uncertain categories. Specifically, we solve the following optimisation problem to identify the uncertain videos.

$$\underset{\mathbf{y},\mathbf{z}}{\text{argmax}} \quad \sum a_i y_i + \sum b_j z_j \tag{7}$$

$$\text{s.t.} \quad \sum z_j = P \tag{8}$$

$$\sum z_j \geq y_i \quad \forall i \text{ connected to } j \tag{9}$$

$$0 \leq z_j \leq 1 \tag{10}$$

$$0 \leq y_i \leq 1 \tag{11}$$

$$\forall i \in I \text{ and } j \in J$$

where I is the set of indices of categories and J is the set of indices of videos.

a_i is the gain associated with selecting the i^{th} category C_i. We choose this to be the uncertainty score of C_i.

b_j is the gain associated with selecting the j^{th} video v_j. We choose this to be the uncertainty score of $v_j = f(C_1, \ldots, C_k)$; for some function f of related categories. For example, a simple version of f can be one that chooses the score of the most uncertain category connected to v_j.

$z_j \in \{0, 1\}$ and $y_i \in \{0, 1\}$ will be the integer solution at optimality.

P is the number of videos for which the user is willing to give feedback.

Feedback is sought from the user for the videos with $z_j = 1$. Note that for each video, feedback is sought only for those categories that are identified as the most uncertain in Section 4.1.1.

4.1.3 Localisation based on Feedback

Based on the feedback provided by the user, we learn and update the category-specific SVM models. For instance, the user may mark category C_i as 'relevant' for video v and C_j as 'irrelevant'. We subsequently treat video v as a positive example for C_i and as a negative example for C_j. We update

Svm_{C_i} and Svm_{C_j} by including v as an additional training example. This gives us updated SVM parameters $\left(\mathbf{w}_{C_i}, b_{C_i}\right)$ and $\left(\mathbf{w}_{C_j}, b_{C_j}\right)$ which we incorporate in the SVM decision functions (the dynamic node features described in Section 3.3.1) for subsequent categorisation. The SVM features are recomputed in the AMN weights as the individual SVM models mature and start stabilising.

4.1.4 Estimating γ_i

The parameter γ_i controls the importance to be associated with the SVM node features. We define γ_i to be:

$$\gamma_i = \begin{cases} 0 & \text{if } confidence\left(Svm_{C_i}\right) < \mathcal{T} \\ confidence\left(Svm_{C_i}\right) & \text{otherwise} \end{cases},$$

where $\mathcal{T}$ is the user-defined threshold, $confidence\left(Svm_{C_i}\right)$ is the $\xi\alpha - estimator$ of F1 score of Svm_{C_i} computed as in [7]. These estimators are developed based on the idea of leave-one-out estimation of the error rate. However, leave-one-out or cross validation estimation is a computationally intensive process. On the other hand, $\xi\alpha - estimator$ can be computed at essentially no extra cost immediately after training every Svm_{C_i} using the slack variables ξ_i in the SVM primal objective and α_i Lagrange variables in the SVM dual formulation.

4.1.5 Category constraints

In the process of localising the GCC, users can indicate (through feedback) that a category C_i suggested by the system is 'completely irrelevant' to the domain. The system remembers this feedback as a *hard constraint*. Thereafter, even if the machine learning model finds this category to be relevant to the input video, it will not be suggested as a candidate for that video.

When a user marks category C_i as completely irrelevant, the system also suggests a set of categories that are *related* to category C_i as candidates for being marked as completely irrelevant. Users may inspect (and modify) this list to quickly discover other uninteresting categories to incorporate them into additional hard constraints.

By *hard constraint* for a category C_i, we mean inference subject to a constraint set that includes $y_i^0 = 1$. If categories C_i and C_j are *related*, we would expect the effect of this constraint to propagate from C_i to C_j and encourage y_j^0 to also become 1.

4.2 Domain classifier

When a category outside the domain of interest is assigned to a video, users can mark 'completely irrelevant' for it during feedback. It is also possible that such a category is not chosen for feedback at all by the active learning algorithm. Such assignments may not be acceptable in a production deployment.

An SVM classifier Svm_{agri}, if enabled, is used as a domain classifier to check the membership of categories in the agriculture domain in such a case. This classification is performed before an AMN inference, on the basis of the category descriptions. Categories marked as 'completely irrelevant' during previous feedbacks are treated as negative examples, whereas, those marked as 'relevant' or 'possibly relevant' are considered positive examples for Svm_{agri}. Categories that are classified as non-agriculture are considered to be 'completely irrelevant' for that inference, essentially producing *dynamic* hard constraints.

Let $confidence\left(Svm_{agri}\right)$ be the $\xi\alpha - estimator$ of F1 score of Svm_{agri}, as in [7]. If $confidence\left(Svm_{agri}\right)$ is less than a user-defined threshold $\mathcal{T}_{agri}$, dynamic hard constraints are not considered.

5. EXPERIMENTS

We performed experiments on our system in a cold-start setting, where we assumed no previous hints at the categorisation for videos that the classifier could use. Due to the highly skewed frequency distribution of the YouTube tags as shown in Figure 1, where most of the tags never get reused, it is difficult to use tags as a basis for warm-start in our system. Using the YouTube categories for warm-start would only result in very high-level hints, which would not be of any use for our purpose of fine-grained categorisation in a particular domain.

We ran our experiments on a more specific subset of the agricultural videos we collected. We chose 741 videos pertaining to *biodiversity* in the agriculture domain, to demonstrate our system's ability to assign facets that relate to specific varieties of plants, fruits, and vegetables according to the input video. Our dataset contains videos ranging from information on growing a particular species of crop such as a Persian Mulberry tree, to a summary of various fruits, such as Thai fruits.

We trained our system with 45 batches of feedback. Each batch provided to the system consisted of 14 videos, and user feedback was solicited for the 7 most uncertain categories each for 7 most uncertain videos in a batch. An example of categorisation done by our system after the 45 batches of feedback is demonstrated in Table 4. Further training would help improve the facet discovery done by the categorisation system.

Title:	The best way to sprout strawberry seeds / growing strawberries
Description:	I find this method has the highest germination success. BTW, If I didn't mention, after you get the sprout into your cup of dirt, gently mist it a few times a week so it doesn't dry up. Good luck!
YouTube categ.:	Education
YouTube tags:	SEEDS, Best Way, Garden Strawberry, Seed, garden, farming, microfarming, strawberry, plants, germinating, germination, sowing
Our facets:	Strawberry, Fragaria, Seed, Sprouting, Plant propagation, Gardening, Seed drill, Musk strawberry, Fragaria vesca, Seed dispersal, Seed saving, Seed enhancement

Table 4: Metadata and assigned facets of a video as an example.

We ran four batches of categorisation on the entire set of biodiversity videos, once with no training at all, and thrice after 15 batches of feedback. We refer to the state of the system at these four points as *Epochs 0* through *3*.

5.1 Model evaluation

The facets generated by our system have less granularity and noise as compared to YouTube tags, while staying sufficiently specific. A comparison of the frequency distributions of YouTube tags and facets generated by our system is shown in Figure 2.

We expect that videos belonging to a similar theme, or regarding a particular type of crop would be assigned similar facets. Following this intuition, we chose cluster quality as a measure to evaluate a categorisation on a collection of videos. We consider the tags/facets assigned to a video as features for the clustering algorithm. We find the best clustering for a particular categorisation by grid search on hyperparameters of the clustering algorithm, under reasonable constraints. To evaluate the quality of clustering, we compute the Silhouette Coefficient at each point in the grid search. Finally, we compare the Silhouette coefficients of each categorisation at its best clustering to determine the quality of the categorisation. A higher Silhouette Coefficient relates to a model with better-defined clusters. The score is bounded between -1 for incorrect clustering and $+1$ for highly dense clustering. Scores around zero indicate overlapping clusters. We compare the clustering quality on the categorisation of YouTube tags and the facets generated by our system in all epochs using DBSCAN[6] and K-Means for clustering.

The clustering analysis using DBSCAN can be seen in Table 5. 'Samples' are the minimum neighbours required within ϵ distance of a sample for it to be considered a core sample. We report the clustering analysis using K-Means in Table 6. In both cases, we have limited the minimum number of clusters to 4.

Facets	ϵ	Samples	#Clusters	Silhouette
YT Tags	2.0	5	5	-0.03386
Epoch 0	1.75	3	4	0.13735
Epoch 1	1.75	3	4	0.14338
Epoch 2	1.75	3	5	0.15049
Epoch 3	2.0	4	4	0.21063

Table 5: DBSCAN cluster analysis.

Facets	#Clusters	Silhouette
YT Tags	4	0.16685
Epoch 0	4	0.21592
Epoch 1	6	0.15296
Epoch 2	4	0.26736
Epoch 3	4	0.34917

Table 6: K-Means cluster analysis.

5.2 Feedback

During our experiments, we provided feedback for 45 batches, each batch with a total of 49 uncertain categories to review (7 uncertain videos with 7 uncertain categories each). Reviewing uncertain categories as 'relevant' or 'irrelevant' forms the training data for the per-category SVMs. We observed that as the system learns, it becomes more difficult to indicate absolute relevancy of an uncertain category, and we see an increase in the number of 'possibly relevant' categories with feedbacks. Simultaneously, as we narrow down on the agriculture domain, the 'completely irrelevant' categories seem to decrease over time. These traits can be visualised in Figure 3.

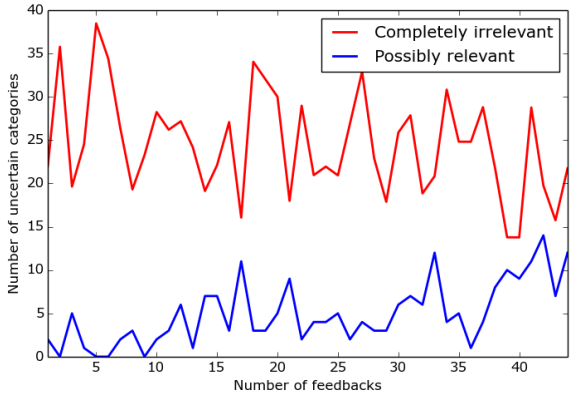

Figure 3: Distribution of completely irrelevant and possibly relevant facets over feedback batches.

5.3 Categorisation and feedback timing

The experiments were run on an Ubuntu server running on a 20-core Intel(R) Xeon(R) CPU E5-2660 v2 @ 2.20GHz with hyper-threading. The amount of RAM available to the experiments was 60 GB.

A plot of the time taken by our system to categorise an individual video in a batch is illustrated in Figure 4. On an average, it takes 1 minute 45 seconds for our system to discover facets for a video.

For the 45 feedback batches, it took an average of 1 minute 10 seconds per video of training the system after a review of 7 videos in a batch, as indicated in Figure 5. However, since our implementation of AMN training is not incremental, it required re-training of the AMN model parameters based on previous feedbacks during each training. We also required re-running of the AMN inference using the category-specific SVMs just trained, while performing AMN training. This added an amount of time to the training process proportional to the facet discovery time for the batch.

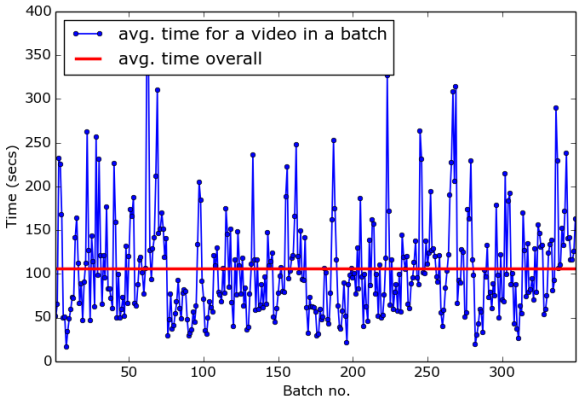

Figure 4: Time taken for categorisation.

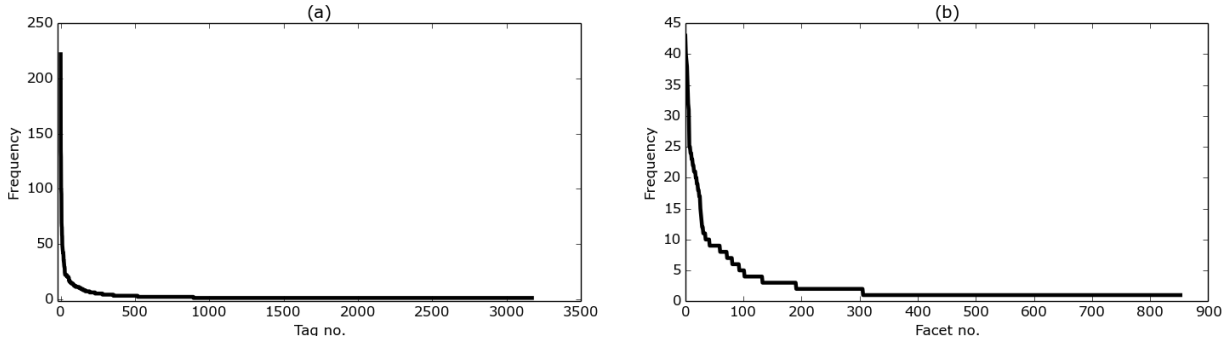

Figure 2: *(a)* Frequency distribution of YouTube tags in agricultural (biodiversity sub-domain) videos from our collection, *(b)* Frequency distribution of facets for agricultural (biodiversity sub-domain) videos in our collection.

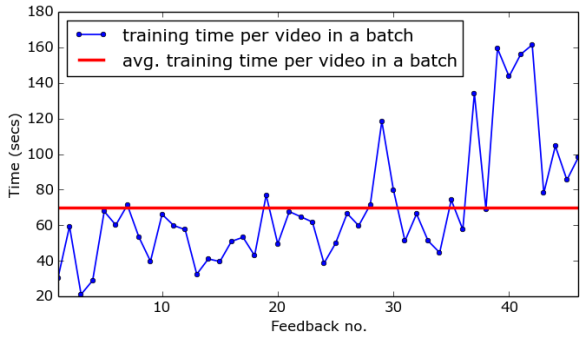

Figure 5: Time taken for training the system after every feedback for a batch.

6. CONCLUSION

We presented our approach to evolve a domain-specific multi-label video categorisation system by adapting a large global category catalogue (Wikipedia) to assign agriculture-related categories in a local video collection. We showed that there is a more desirable balance between granularity and cardinality of these categories as compared to that of YouTube's tags. By staying specific to the agriculture domain and narrowing down on a local category catalogue for the same, we also cater to the perceptions of the farmer groups, who are the intended beneficiaries of our work.

Exploring the possibilities of avoiding the retraining of the AMN model parameters after every feedback, that is, making the learning fully incremental, forms a part of our future work. We also plan to delve into additional active learning methods for the system, aiming toward minimised cognitive load on the user and a better learning rate. Additionally, we will incorporate more features for categorisation of videos from YouTube, such as comments, audio-to-text, and the like.

Our vision also includes figuring out the most effective way for farmer groups to utilise this video data in the form of a mobile application, and the development of such an application.

7. REFERENCES

[1] M. R. Amer, E. Bilgazyev, S. Todorovic, S. K. Shah, I. A. Kakadiaris, and L. Ciannelli. Fine-grained categorization of fish motion patterns in underwater videos. In *ICCV Workshops*, pages 1488–1495. IEEE, 2011.

[2] R. B. Bairi and G. Ramakrishnan. Labeling documents in search collection: Evolving classifiers on a semantically relevant label space. In *Proceedings of Workshop on Semantic Matching in Information Retrieval, 2014*, 2014.

[3] R. B. Bairi, G. Ramakrishnan, and V. Sindhwani. Personalized classifiers: Evolving a classifier from a large reference knowledge graph. In *Proceedings of IDEAS '14 July 07-09 2014*, 2014.

[4] 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.

[5] Y. Dong, J. Zhang, X. Chang, and J. Zhao. Automatic sports video genre categorization for broadcast videos. In *VCIP*, pages 1–5. IEEE, 2012.

[6] M. Ester, H. peter Kriegel, J. S, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996.

[7] T. Joachims. Estimating the generalization performance of an svm efficiently. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 431–438, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[8] D. Milne. An open-source toolkit for mining wikipedia. In *In Proc. New Zealand Computer Science Research Student Conf*, page 2009.

[9] 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.

[10] Y. Song, M. Z. 0003, J. Yagnik, and X. Wu. Taxonomic classification for web-based videos. In *CVPR*, pages 871–878. IEEE, 2010.

[11] B. Taskar, V. Chatalbashev, and D. Koller. Learning associative markov networks. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, pages 102–, New York, NY, USA, 2004. ACM.