Learning from Video and Text via Large-Scale Discriminative Clustering

Antoine Miech\textsuperscript{1,2}, Jean-Baptiste Alayrac\textsuperscript{1,2}, Piotr Bojanowski\textsuperscript{2}, Ivan Laptev\textsuperscript{1,2}, Josef Sivic\textsuperscript{1,2,3}
\textsuperscript{1}École Normale Supérieure \textsuperscript{2}Inria \textsuperscript{3}CIIRC

Abstract

Discriminative clustering has been successfully applied to a number of weakly supervised learning tasks. Such applications include person and action recognition, text-to-video alignment, object co-segmentation and co-localization in videos and images. One drawback of discriminative clustering, however, is its limited scalability. We address this issue and propose an online optimization algorithm based on the Block-Coordinate Frank-Wolfe algorithm. We apply the proposed method to the problem of weakly supervised learning of actions and actors from movies together with corresponding movie scripts. The scaling up of the learning problem to 66 feature-length movies enables us to significantly improve weakly supervised action recognition.

1. Introduction

Action recognition has been significantly improved in recent years. Most existing methods \cite{23, 35, 39, 40} rely on supervised learning and, therefore, require large-scale, diverse and representative action datasets for training. Collecting such datasets, however, is a difficult task given the high costs of manual search and annotation of the video. Notably, the largest action datasets today are still orders of magnitude smaller (UCF101 \cite{37}, ActivityNet \cite{7}) compared to large image datasets, they often contain label noise and target specific domains such as sports (Sports1M \cite{20}).

Weakly supervised learning aims to bypass the need of manually-annotated datasets using readily-available, but possibly noisy and incomplete supervision. Examples of such methods include learning of person names from image captions or video scripts \cite{3, 10, 36, 38}. Learning actions from movies and movie scripts has been approached in \cite{4, 5, 9, 23}. Most of the work on weakly supervised person and action learning, however, has been limited to one or a few movies. Therefore the power of leveraging large-scale weakly annotated video data has not been fully explored.

In this work we aim to scale weakly supervised learning of actions. In particular, we follow the work of \cite{4} and learn actor names together with their actions from movies and movie scripts. While actors are learned separately for each movie, differently from \cite{4}, our method simultaneously learns actions from all movies and movie scripts available for training. Such an approach, however, requires solving a large-scale optimization problem. We address this issue and propose to scale weakly supervised learning by adapting the Block-Coordinate Frank-Wolfe approach \cite{21}. Our optimization procedure enables action learning from tens of movies and thousands of action samples, readily available from our subset of movies or other recent datasets with movie descriptions \cite{31}. This, in turn, results in large improvements in action recognition.

Besides the optimization, our work introduces a new model for background class in the form of a constraint. It enables better and automatic modeling of the background class (i.e. unknown actors and actions). We evaluate our method on 66 movies and demonstrate significant improvements for both actor and action recognition. Example results are illustrated in Figure 1.
1.1. Related Work

This section reviews related work on discriminative clustering, Frank-Wolfe optimization and its applications to the weakly supervised learning of people and actions in video.

**Discriminative clustering and Frank-Wolfe.** The Frank-Wolfe algorithm [11, 15] allows to minimize large convex problems over convex sets by solving a sequence of linear problems. In computer vision, it has been used in combination with discriminative clustering [2] for action localization [5], text-to-video alignment [1, 6], object co-localization in videos and images [18] or instance-level segmentation [32]. A variant of Frank-Wolfe with randomized block coordinate descent was proposed in [21]. This extension leads to lower complexity in terms of time and memory requirements while preserving the convergence rate. In this work we build on [21] and adapt it for the problem of large-scale weakly supervised learning of actions from movies.

**Weakly supervised action recognition.** Movie scripts are used as a source of weak supervision for temporal action localization in [9]. An extension of this work [5] exploits the temporal order of actions as a learning constraint. Other [22] target spatio-temporal action localization and recognition in video using a latent SVM. A weakly supervised extension of this method [34] localizes actions without location supervision at the training time. Another recent work [41] proposes a multi-fold Multiple-Instance Learning (MIL) SVM to localize actions given video-level supervision at training time. Closer to us is the work of [4] that improves weakly supervised action recognition by joint action-actor constraints derived from scripts. While the approach in [4] is limited to a few action classes and movies, we propose here a scalable solution and demonstrate significant improvements in action recognition when applied to the large-scale weakly supervised learning of actions from many movies.

**Weakly supervised person recognition.** Person recognition in TV series has been studied in [10, 36] where the authors propose a solution to the problem of associating speaker names in scripts and faces in videos. Speakers in videos are identified by detecting face tracks with lip motion. The method in [8] presents an alternative solution by formulating the association problem using a convex surrogate loss. Parkhi et al. [28] present a method for person recognition combining a MIL SVM with a model for the background class. Most similar to our model is the one presented in [4]. The authors propose a discriminative clustering cost under linear constraints derived from scripts to recover the identities and actions of people in movies. Apart from scaling-up the approach of [4] to much larger datasets, our model extends and improves [4] with a new background constraint.

**Contributions.** In this paper we make the following contributions: (i) We propose an optimization algorithm based on Block-Coordinate Frank-Wolfe that allows scaling up discriminative clustering models [2] to much larger datasets. (ii) We extend the joint weakly supervised Person-Action model of [4], with a simple yet efficient model of the background class. (iii) We apply the proposed optimization algorithm to scale-up discriminative clustering to an order of magnitude larger dataset, resulting in significantly improved action recognition performance.

2. Discriminative Clustering for Weak Supervision

We want to assign labels (e.g. names or action classes) to samples (e.g. person tracks in the video). As opposed to the standard supervised learning setup, the exact labels of samples are not known at training time. Instead, we are given only partial information that some samples in a subset (or bag) may belong to some of the labels. This ambiguous setup, also known as multiple instance learning, is common, for example, when learning human actions from videos and associated text descriptions.

To address this challenge of ambiguous and partial labels, we build on the discriminative clustering criterion based on a linear classifier and a quadratic loss (DIFFRAC [2]). This framework has shown promising results in weakly supervised and unsupervised computer vision tasks [1, 4, 5, 6, 16, 17, 30, 32]. In particular, we use this approach to group samples into linearly separable clusters. Suppose we have \( N \) samples to group into \( K \) classes. We are given \( d \)-dimensional features \( X \in \mathbb{R}^{N \times d} \), one for each of the \( N \) samples, and our goal is to find a binary matrix \( Y \in \{0, 1\}^{N \times K} \) assigning each of the \( N \) samples to one of the labels, where \( Y_{nk} = 1 \) if and only if the sample \( n \) (e.g. a person track in a movie) is assigned to the label \( k \) (e.g. action class running).

First, suppose the assignment matrix \( Y \) is given. In this case finding a linear classifier \( W \) can be formulated as a ridge regression problem

\[
\min_{W \in \mathbb{R}^{d \times K}} \frac{1}{2N} \| Y - XW \|_F^2 + \frac{\lambda}{2} \| W \|_F^2, \tag{1}
\]

where \( X \) is a matrix of input features, \( Y \) is the given labels assignment matrix, \( \| . \|_F \) is the matrix norm (or Frobenius norm) induced by the matrix inner product \( \langle . , . \rangle_F \) (or Frobenius inner product) and \( \lambda \) is a regularization hyperparameter set to a fixed constant. The key observation is that we can resolve the classifier \( W^* \) in closed form as

\[
W^*(Y) = (X^\top X + N\lambda I)^{-1}X^\top Y. \tag{2}
\]

In our weakly supervised setting, however, \( Y \) is unknown. Therefore, we treat \( Y \) as a latent variable and optimize (1) w.r.t. \( W \) and \( Y \). In details, plugging the optimal
solution $W^*$ (2) in the cost (1) removes the dependency on $W$ and the cost can be written as a quadratic function of $Y$, i.e. $C(Y) = (Y, A(X, \lambda)Y)_F$, where $A(X, \lambda)$ is a matrix that only depends on the data $X$ and a regularization parameter $\lambda$. Finding the best assignment matrix $Y$ can then be written as the minimization of the cost $C(Y)$:

$$\min_{Y \in \{0,1\}^{N\times K}} (Y, A(X, \lambda) Y)_F. \quad (3)$$

Solving the above problem, however, can lead to degenerate solutions [2] unless additional constraints on $Y$ are provided. In section 3, we incorporate weak supervision in the form of constraints on the latent assignment matrices $Y$. The constraints on $Y$ used for weak supervision generally decompose into small independent blocks. This block structure is the key for our optimization approach that we will present next.

### 2.1. Large-Scale optimization

The Frank-Wolfe (FW) algorithm has been shown effective for optimizing convex relaxation of (3) in a number of vision problems [1, 4, 5, 6, 19, 32]. It only requires solving linear programs on a set of constraints. Therefore, it avoids costly projections and allows the use of complicated constraints such as temporal ordering [5]. However, the standard FW algorithm is not well suited to solve (3) for a large number of samples $N$.

First, storing the $N \times N$ matrix $A(X, \lambda)$ in memory becomes prohibitive (e.g. the size of $A$ becomes $\geq 100$GB for $N \geq 200$000). Second, each update of the FW algorithm requires a full pass over the data resulting in a space and time complexity of order $N$ for each FW step.

Weakly supervised learning is, however, largely motivated by the desire of using large-scale data with “cheap” and readily-available but incomplete and noisy annotation. Scaling up weakly supervised learning to a large number of samples is, therefore, essential for its success. We address this issue and develop an efficient version of the FW algorithm. Our solution builds on the Block-Coordinate Frank-Wolfe (BCFW) [21] algorithm and extends it with a smart block-dependent update procedure as described next. The proposed update procedure is one of the key contributions of this paper.

#### 2.1.1 Block-coordinate Frank-Wolfe (BCFW)

The Block-Coordinate version of the Frank-Wolfe algorithm [21] is useful when the convex feasible set $\mathcal{Y}$ can be written as a Cartesian product of $n$ smaller sets of constraints: $\mathcal{Y} = \mathcal{Y}^{(1)} \times \ldots \times \mathcal{Y}^{(n)}$. Inspired by coordinate descent techniques, BCFW consists of updating one variable block $\mathcal{Y}^{(i)}$ at a time with a reduced Frank-Wolfe step. This method has potentially $n$ times cheaper iterates both in space and time. We will see that most of the weakly supervised problems exhibit such a block structure on latent variables.

#### 2.1.2 BCFW for discriminative clustering

To benefit from BCFW, we have to ensure that the time and space complexity of the block update does not depend on the total number of samples $N$ (e.g. person tracks in all movies) but only depends on the size $N_i$ of smaller blocks of samples $i$ (e.g. person tracks within one movie). After a block is sampled, the update consists of two steps. First, the gradient with respect to the block is computed. Then the linear oracle is called to obtain the next iterate. As we show below, the difficult part in our case is to efficiently compute the gradient with respect to the block.

**Block gradient: a naive approach.** Let’s denote $N_i$ the size of block $i$. The objective function $f$ of problem (3) is $f(Y) = (Y, A(X, \lambda) Y)_F$, where (see [2])

$$A(X, \lambda) = \frac{1}{2N} (I_N - X(X^\top X + N \lambda I_d)^{-1} X^\top). \quad (4)$$

To avoid storing matrix $A(X, \lambda)$ of size $N \times N$, one can pre-compute the matrix $P = (X^\top X + N \lambda I_d)^{-1} X^\top \in \mathbb{R}^{d \times N}$. We can write the block gradient with respect to a subset of samples $i$ as follows:

$$\nabla_{(i)} f(Y) = \frac{1}{N} (Y^{(i)} - X^{(i)} PY), \quad (5)$$

where $Y^{(i)} \in \mathbb{R}^{N_i \times K}$ and $X^{(i)} \in \mathbb{R}^{N_i \times d}$ are the label assignment variable and the feature matrix for block $i$ (e.g. person tracks in movie $i$), respectively. Because of the $PY$ matrix multiplication, naively computing this formula has the complexity $O(NdK)$, where $N$ is the total number of samples, $d$ is the dimensionality of the feature space and $K$ is the number of classes. As this depends linearly on $N$, we aim to find a more efficient way to compute block gradients, as described next.

**Block gradient: a smart update.** We now propose an update procedure that avoids re-computation of block gradients and whose time and space complexity at each iteration depends on $N_i$ instead of $N$. The main intuition is that we need to find a way to store information about all the blocks in a compact form. A natural way of doing so is to maintain the weights of the linear regression parameters $W \in \mathbb{R}^{d \times K}$. From (2) we have $W = PY$. If we are able to maintain the variable $W$ at each iteration with the desired complexity $O(N_dK)$, then the block gradient computation (5) can be reduced from $O(N_dK)$ to $O(N_i dK)$. We now explain how to effectively achieve that.
At each iteration $t$ of the algorithm, we only update a block $i$ of $Y$ while keeping all other blocks fixed. We denote the direction of the update by $D_i \in \mathbb{R}^{N \times K}$ and the step size by $\gamma_t$. With this notation the update becomes

$$Y_{t+1} = Y_t + \gamma_tD_t,$$

(6)

The update rule for the weight variable $W_t$ can now be written as follows:

$$W_{t+1} = P(Y_t + \gamma_tD_t)$$

$$W_{t+1} = W_t + \gamma_tPD_t,$$

(7)

Recall that at iteration $t$, BCFW only updates block $i$, therefore $D_t$ has non zero value only at block $i$. In block notation we can therefore write the matrix product $PD_t$ as:

$$[P^{(1)}, \ldots, P^{(i)}, \ldots, P^{(n)}] \times \begin{bmatrix} 0 \\ D_t^{(i)} \\ 0 \\ \vdots \\ 0 \end{bmatrix} = P^{(i)}D_t^{(i)},$$

where $P^{(i)} \in \mathbb{R}^{d \times N_i}$ and $D_t^{(i)} \in \mathbb{R}^{N_i \times K}$ are the $i$-th blocks of matrices $P$ and $D_t$, respectively. The outcome is an update of the following form

$$W_{t+1} = W_t + \gamma_tP^{(i)}D_t^{(i)},$$

(9)

where the computational complexity for updating $W$ has been reduced to $O(N_idK)$ compared to $O(NdK)$ in the standard update.

We have designed a Block-Coordinate Frank-Wolfe update with time and space complexity depending only on the size of the blocks and not the entire dataset. This allows to scale discriminative clustering to problems with a very large number of samples. The pseudo-code for the algorithm is summarized in Algorithm 1. Next, we describe an application of this large-scale discriminative clustering algorithm to weakly supervised person and action recognition in movies.

### 3. Weakly supervised Person-Action model

We now describe an application of our large-scale discriminative clustering algorithm with weak-supervision. The goal is to assign to each person track a name and an action. Both names and actions are mined from movie scripts. For a given movie $i$, we assume to have $N_i$ automatically extracted person tracks as well as the parsing of a movie script into person names and action classes. We also assume that scripts and movies have been roughly aligned in time. In such a setup we can assign labels (e.g. a name or an action) from a script section to a subset of tracks $\mathcal{N}$ from the corresponding time interval of a movie (see Figure 2 for example). In the following, we explain how to convert such form of weak supervision into a set of constraints on latent variables corresponding to the names and actions of people. We will also show how these constraints easily decompose into blocks. We denote $Z$ the latent variable assignment matrix for person names and $T$ for actions.

#### 3.1. Weak-supervision as constraints

We use linear constraints to incorporate weak supervision from movie scripts. In detail, we define constraints on subsets of person tracks that we call “bags”. In the following we explain the procedure for construction of bags together with the definition of the appropriate constraints.

‘At least one’ constraint. Suppose a script reveals the presence of a person $p$ in some time interval of the movie. We construct a set $\mathcal{N}$ with all person tracks in this interval. As first proposed by [4], we model that at least one track in $\mathcal{N}$ is assigned to person $p$ by the following constraint

$$\sum_{n \in \mathcal{N}} Z_{np} \geq 1.$$  

(10)

We can apply the same type of constraint when solving for action assignment $T$.

**Person-Action constraint.** Scripts can also provide information that a person $p$ is performing an action $a$ in a scene. In such cases we can formulate stricter and more informative constraints as follows. We construct a set $\mathcal{N}$ containing all person tracks appearing in this scene. Following [4], we formulate a joint constraint on presence of a person performing a specific action as

$$\sum_{n \in \mathcal{N}} Z_{np}T_{na} \geq 1.$$  

(11)

**Mutual exclusion constraint.** We also model that each person track can only be assigned to exactly one label. This

### Algorithm 1 BCFW for Discriminative Clustering [2]

Initiate $Y_0$, $P := (X^T X + N\lambda I_d)^{-1}X^T$, $W_0 = PY_0$, $g_i = +\infty$, $\forall i$.

for $t = 1 \ldots N_{iter}$ do

1. $i \leftarrow$ sample from distribution proportional to $g$ [27]
2. $\nabla_{(i)} f(Y_t) \leftarrow \frac{1}{N}(Y_t^{(i)} - X^{(i)}W_t)$ # Block gradient
3. $Y_{min} \leftarrow \arg \min_{y \in Y(i)} \nabla_{(i)} f(Y_t), x_F$ # Linear oracle
4. $D \leftarrow Y_{min} - Y(i)$
5. $g_i \leftarrow -\langle D, \nabla_{(i)} f(Y_t) \rangle_F$ # Block gap
6. $\gamma \leftarrow \min(1, \frac{g_i}{\nabla_{(i)} f(Y_t), x_F})$ # Line-search
7. $W_{t+1} = W_t + \gamma_p(i) D$ # W update
8. $Y_{t+1} \leftarrow Y_t + \gamma D$ # Block update

end for
restriction can be formalized by the mutual exclusion constraint

\[ Z1_P = 1_N, \]  

(12)

for \( Z \) (i.e. rows sum up to 1). Same constraint holds for \( T \).

**Background class constraint.** One of our contributions is a novel way of coping with the background class. As opposed to previous work [4], our approach allows us to have background model that does not require any external data. Also it does not require a specific background class classifier as in [28].

Our background class constraint can be seen as a way to supervise people and actions that are not mentioned in scripts. We observe that tracks that are not subject to constraints from Eq. (10) and tracks that belong to crowded shots are likely to belong to the background class. Let us denote by \( B \) the set of such tracks. We impose that at least a certain fraction \( \alpha \in [0, 1] \) of tracks in \( B \) must belong to the background class. Assuming that person label \( p = 1 \) corresponds to the background, we obtain the following linear constraint (similar constraint can be defined for actions on \( T \)):

\[ \sum_{n \in B} Z_{n1} \geq \alpha |B|. \]  

(13)

**3.2. Person-Action model formulation**

Here we summarize the complete formulation of the person and action recognition problems.

**Solving for names.** We formulate the person recognition problem as discriminative clustering, where \( X_1 \) are face descriptors:

\[ \min_{Z \in \{0,1\}^{N \times P}} (Z, A(X_1, \lambda)Z)_F, \quad \text{(Discriminative cost)} \]  

(14)

such that

\[ \begin{cases} \sum_{n \in N} Z_{np} \geq 1, & \text{(At least one)} \\ \sum_{n \in B} Z_{n1} \geq \alpha |B|, & \text{(Background)} \\ Z1_P = 1_N, & \text{(Mutual exclusion)} \end{cases} \] 

**Solving for actions.** After solving the previous problem for names separately for each movie, we vertically concatenate all person name assignment matrices \( Z \). We also define a single action assignment variable \( T \) in \( \{0,1\}^{M \times A} \), where \( M \) is the total number of tracks across all movies and \( X_2 \) are action descriptors (details given later). We formulate our action recognition problem as a large QP:

\[ \min_{T \in \{0,1\}^{M \times A}} \langle T, A(X_2, \mu)T \rangle_F, \quad \text{(Discriminative cost)} \]  

(15)

such that

\[ \begin{cases} \sum_{n \in N} T_{na} \geq 1, & \text{(At least one)} \\ \sum_{n \in N} Z_{np} T_{na} \geq 1, & \text{(Person-Action)} \\ \sum_{n \in B} T_{n1} \geq \beta |B|, & \text{(Background)} \\ T1_A = 1_M, & \text{(Mutual exclusion)} \end{cases} \] 

**Block-Separable constraints.** The set of linear constraints on the action assignment matrix \( T \) is block separable since each movie has its own set of constraints, i.e. there are no constraints spanning multiple movies. Therefore, we can fully demonstrate here the power of our large-scale discriminative clustering optimization (Algorithm 1).

**4. Experimental Setup**

**4.1. Dataset**

Our dataset is composed of 66 Hollywood feature-length movies (see the list in Appendix [26]) that we obtained from either BluRay or DVD. For all movies, we downloaded their scripts (on www.dailyscript.com) that temporally aligned with the videos and movie subtitles using the method described in [23]. The total number of frames in all 66 movies is 11,320,252. The number of body tracks detected across all movies (see 4.3 for more details) is \( M = 201874 \).

**4.2. Text pre-processing**

To provide weak supervision for our method we process movie scripts to extract occurrences of the 13 most frequent action classes: Stand Up, Eat, Sit Down, Sit Up,
illustrates our qualitative results for our approach to shows the sensitivity to hyper-parameter shows the Accuracy (Acc.) and Average appearance of each body track we extract ResNet-50 \[ the HOF, MBHx and MBHy channels. In order to capture trajectory descriptors \[ compute bag-of-visual-words representation of dense trajectory. Such tracks can be originating from any actor in the may look away from the camera, we cannot obtain its identity. For each detected face, we compute facial landmarks \[ follow the face alignment and resizing of face images to 224x224 pixel. We use pre-trained vgg-face features \[ to extract descriptors for each face. We kept the features of dimension 4096 computed by the network at the last fully-connected layer that we \[ normalized. For each face track, we choose the top K (in practice, we choose K=5) faces that have the best facial landmark confidence. Then we represent each track by averaging the features of the top K faces.

**Face tracks.** To obtain tracks of faces in the video, we run the multi-view face detector [25] based on the DPM model [12]. We then extract face tracks using the same method as in [10, 36]. For each detected face, we compute facial landmarks [36] followed by the face alignment and resizing of face images to 224x224 pixel. We use pretrained vgg-face features [29] to extract descriptors for each face. We kept the features of dimension 4096 computed by the network at the last fully-connected layer that we \[ normalized. For each face track, we choose the top K (in practice, we choose K=5) faces that have the best facial landmark confidence. Then we represent each track by averaging the features of the top K faces.

**Body tracks.** To get the person body tracks, we run the Faster-RCNN network with VGG-16 architecture fine-tuned on VOC 07 [33]. Then we track bounding boxes using the same tracker as used to obtain face tracks. To get person identity for body tracks, we greedily link each body track to one face track by maximizing a spatio-temporal bounding box overlap measure. However if a body track does not have an associated face track as the actor’s face may look away from the camera, we cannot obtain its identity. Such tracks can be originating from any actor in the movie. To capture motion features of each body track, we compute bag-of-visual-words representation of dense trajectory descriptors [39] inside the bounding boxes defined by the body track. We use 4000 cluster centers for each of the HOF, MBHx and MBHy channels. In order to capture appearance of each body track we extract ResNet-50 [13] pre-trained on ImageNet. For each body bounding box, we compute the average RoI-pooled [33] feature map of the last convolutional layer within the bounding box, which yields a feature vector of dimension 2048 for each box. We extract a feature vector every 10th frame, average extracted feature vectors over the duration of the track and \[ normalize. Finally, we concatenate the dense trajectory descriptor and the appearance descriptor resulting in a 14028-dimensional descriptor for each body track.

### 4.3. Person detection and Features

#### Face tracks.

To obtain tracks of faces in the video, we run the multi-view face detector [25] based on the DPM model [12]. We then extract face tracks using the same method as in [10, 36]. For each detected face, we compute facial landmarks [36] followed by the face alignment and resizing of face images to 224x224 pixel. We use pretrained vgg-face features [29] to extract descriptors for each face. We keep the features of dimension 4096 computed by the network at the last fully-connected layer that we \[ normalized. For each face track, we choose the top K (in practice, we choose K=5) faces that have the best facial landmark confidence. Then we represent each track by averaging the features of the top K faces.

#### Body tracks.

To get the person body tracks, we run the Faster-RCNN network with VGG-16 architecture fine-tuned on VOC 07 [33]. Then we track bounding boxes using the same tracker as used to obtain face tracks. To get person identity for body tracks, we greedily link each body track to one face track by maximizing a spatio-temporal bounding box overlap measure. However if a body track does not have an associated face track as the actor’s face may look away from the camera, we cannot obtain its identity. Such tracks can be originating from any actor in the movie. To capture motion features of each body track, we compute bag-of-visual-words representation of dense trajectory descriptors [39] inside the bounding boxes defined by the body track. We use 4000 cluster centers for each of the HOF, MBHx and MBHy channels. In order to capture appearance of each body track we extract ResNet-50 [13] pre-trained on ImageNet. For each body bounding box, we compute the average RoI-pooled [33] feature map of the last convolutional layer within the bounding box, which yields a feature vector of dimension 2048 for each box. We extract a feature vector every 10th frame, average extracted feature vectors over the duration of the track and \[ normalize. Finally, we concatenate the dense trajectory descriptor and the appearance descriptor resulting in a 14028-dimensional descriptor for each body track.

### 4.3. Person detection and Features

**Face tracks.** To obtain tracks of faces in the video, we run the multi-view face detector [25] based on the DPM model [12]. We then extract face tracks using the same method as in [10, 36]. For each detected face, we compute facial landmarks [36] followed by the face alignment and resizing of face images to 224x224 pixel. We use pretrained vgg-face features [29] to extract descriptors for each face. We keep the features of dimension 4096 computed by the network at the last fully-connected layer that we \[ normalized. For each face track, we choose the top K (in practice, we choose K=5) faces that have the best facial landmark confidence. Then we represent each track by averaging the features of the top K faces.

**Body tracks.** To get the person body tracks, we run the Faster-RCNN network with VGG-16 architecture fine-tuned on VOC 07 [33]. Then we track bounding boxes using the same tracker as used to obtain face tracks. To get person identity for body tracks, we greedily link each body track to one face track by maximizing a spatio-temporal bounding box overlap measure. However if a body track does not have an associated face track as the actor’s face may look away from the camera, we cannot obtain its identity. Such tracks can be originating from any actor in the movie. To capture motion features of each body track, we compute bag-of-visual-words representation of dense trajectory descriptors [39] inside the bounding boxes defined by the body track. We use 4000 cluster centers for each of the HOF, MBHx and MBHy channels. In order to capture appearance of each body track we extract ResNet-50 [13] pre-trained on ImageNet. For each body bounding box, we compute the average RoI-pooled [33] feature map of the last convolutional layer within the bounding box, which yields a feature vector of dimension 2048 for each box. We extract a feature vector every 10th frame, average extracted feature vectors over the duration of the track and \[ normalize. Finally, we concatenate the dense trajectory descriptor and the appearance descriptor resulting in a 14028-dimensional descriptor for each body track.

### Table 1: Comparison on the Casablanca benchmark [4].

<table>
<thead>
<tr>
<th>Method</th>
<th>Acc.</th>
<th>Multi-Class AP</th>
<th>Background AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cour et al. [8]</td>
<td>48%</td>
<td>63%</td>
<td>-</td>
</tr>
<tr>
<td>Sivic et al. [36]</td>
<td>49%</td>
<td>63%</td>
<td>-</td>
</tr>
<tr>
<td>Bojanowski et al. [4]</td>
<td>57%</td>
<td>75%</td>
<td>51%</td>
</tr>
<tr>
<td>Parkhi et al. [28]</td>
<td>74%</td>
<td>93%</td>
<td>75%</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td><strong>83%</strong></td>
<td><strong>94%</strong></td>
<td><strong>82%</strong></td>
</tr>
</tbody>
</table>

### Table 2: Comparison on the Buffy benchmark [36] using AP.

<table>
<thead>
<tr>
<th>( \alpha )</th>
<th>0</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
<th>0.75</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Accuracy</strong></td>
<td>58</td>
<td>58</td>
<td>70</td>
<td>82</td>
<td>84</td>
<td>83</td>
<td>76</td>
<td>55</td>
</tr>
<tr>
<td><strong>AP</strong></td>
<td>86</td>
<td>87</td>
<td>90</td>
<td>94</td>
<td>94</td>
<td>93</td>
<td>85</td>
<td>58</td>
</tr>
</tbody>
</table>

### Table 3: Sensitivity to hyper-parameter \( \alpha \) (13) on Casablanca.

### 5. Evaluation

#### 5.1. Evaluation of person recognition

We compare our person recognition method to several other methods on the Casablanca benchmark from [4] and the Buffy benchmark from [36]. All methods are evaluated on the same inputs: same face tracks, scripts and characters. Table 1 shows the Accuracy (Acc.) and Average Precision (AP) of our approach compared to other methods on the Casablanca benchmark [4]. In particular we compare to Parkhi et al. [28] which is a strong baseline using the same CNN face descriptors as in our method. We also show the AP of classifying the background character class (Background AP). We compare in Table 2 our approach to other methods [28, 36] reporting results on season 5 of the TV series “Buffy the Vampire Slayer”. Both of these methods [28, 36] use speaker detection to mine additional strong (but possibly incorrect) labels from the script, which we also incorporate (as additional bags) to make the comparison fair. Our method demonstrates significant improvement over the previous results. It also outperforms other methods on the task of classifying background characters. Finally, Table 3 shows the sensitivity to hyper-parameter \( \alpha \) from the background constraint (13) on the Casablanca benchmark. Note that in contrast to other methods, our background model does not require supervision for the background class. This clearly demonstrates the advantage of our proposed background model. For all experiments the hyper-parameter \( \alpha \) of the background constraint (13) was set to 30%. Figure 5 illustrates our qualitative results for character recognition in different movies.
Table 4: Average Precision of actions evaluated on 5 movies. (St.U: Stand Up, E: Eat, S.D: Sit Down, Si.U: Sit Up, H.S: Hand Shake, F: Fight, G.C: Get out of Car, K: Kiss, H: Hug, A: Answer Phone, R: Run, O.D: Open Door, D: Drive)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Random</td>
<td>∅</td>
<td>No</td>
<td>0.9</td>
<td>0.1</td>
<td>0.7</td>
<td>0.1</td>
<td>0.1</td>
<td>0.6</td>
<td>0.2</td>
<td>0.3</td>
<td>0.2</td>
<td>1.8</td>
<td>0.8</td>
<td>0.4</td>
<td>0.5</td>
<td>15.6</td>
</tr>
<tr>
<td>(b) Script only</td>
<td>∅</td>
<td>No</td>
<td>3.0</td>
<td>4.3</td>
<td>5.5</td>
<td>2.8</td>
<td>4.7</td>
<td>2.5</td>
<td>1.6</td>
<td>11.3</td>
<td>4.2</td>
<td>1.4</td>
<td>13.7</td>
<td>3.1</td>
<td>3.0</td>
<td>4.7</td>
</tr>
<tr>
<td>(c) Fully-supervised</td>
<td>4</td>
<td>No</td>
<td>21.2</td>
<td>0.2</td>
<td>22.2</td>
<td>0.9</td>
<td>0.6</td>
<td>7.3</td>
<td>1.4</td>
<td>1.9</td>
<td>4.5</td>
<td>2.0</td>
<td>33.2</td>
<td>18.5</td>
<td>6.3</td>
<td>9.3</td>
</tr>
<tr>
<td>(d) Few training movies</td>
<td>5</td>
<td>Yes</td>
<td>22.6</td>
<td>9.6</td>
<td>15.6</td>
<td>8.1</td>
<td>9.7</td>
<td>6.1</td>
<td>1.0</td>
<td>6.0</td>
<td>2.1</td>
<td>4.2</td>
<td>44.0</td>
<td>16.2</td>
<td>15.9</td>
<td>12.4</td>
</tr>
<tr>
<td>(e) No Joint Model</td>
<td>66</td>
<td>No</td>
<td>10.7</td>
<td>7.0</td>
<td>17.1</td>
<td>7.3</td>
<td>18.0</td>
<td>12.6</td>
<td>2.0</td>
<td>14.9</td>
<td>3.6</td>
<td>5.8</td>
<td>24.4</td>
<td>14.2</td>
<td>24.9</td>
<td>12.5</td>
</tr>
<tr>
<td>(f) Full setup</td>
<td>66</td>
<td>Yes</td>
<td>27.0</td>
<td>9.8</td>
<td>28.2</td>
<td>6.7</td>
<td>7.8</td>
<td>5.9</td>
<td>1.0</td>
<td>12.9</td>
<td>1.7</td>
<td>5.7</td>
<td>56.3</td>
<td>21.3</td>
<td>29.7</td>
<td>16.4</td>
</tr>
</tbody>
</table>

Figure 3: PR curves of action SitDown from Casablanca.

Figure 4: Action recognition mAP with increasing number of training movies.

5.2. Evaluation of action recognition

First, we compare our method to Bojanowski et al. 2013 [4]. Their evaluation uses different body tracks than ours, we design here an algorithm-independent evaluation setup. We compare our model using the Casablanca movie and the Sit Down action. For the purpose of evaluation, we have manually annotated all person tracks in the movie and then manually labeled whether or not they contain the Sit Down action. Given this ground truth, we assess the two models in a similar way as typically done in object detection. Figure 3 shows a precision-recall curve evaluating recognition of the Sit Down action. We show our method trained on Casablanca only (as done in [4]) and then on all 66 movies. Our method trained on Casablanca is already better than [4]. The improvement becomes even more evident when training our method on all 66 movies.

To evaluate our method on our 13 action classes, we use five movies (American Beauty, Casablanca, Double Indemnity, Forrest Gump and Fight Club). For each of these movies we have manually annotated all person tracks produced by our tracker according to 13 target action classes and the background action class. We assume that each track corresponds to at most one target action. In rare cases where this assumption is violated, we annotate the track by one of the correct action classes.

In Table 4 we compare results of our model to different baselines. The first baseline (a) corresponds to the random assignment of action classes. The second baseline (b) Script only uses information extracted from the scripts: each time an action appears in a bag, all person tracks in this bag are then simply annotated with this action. Baseline (c) is using our action descriptors but trained in a fully supervised setup on a small subset of annotated movies. To demonstrate the strength of this baseline we have used the same action descriptors on the LSMDC2016 movie clip retrieval challenge. This is the largest public benchmark [31] related to our work that considers movie data (but without person localization as we do in our work). Table 5 shows our features employed in simple CCA method as done in [24] achieving state-of-the-art on this benchmark. The fourth baseline (d) is our method trained only using the five evaluated movies. The fifth baseline (e) is our model without the joint person-action constraint (11), but still trained on all 66 movies. Finally, the last result (f) is from our model using all the 66 training movies and person-action constraints (11). Results demonstrate that optimizing our model on more movies brings the most significant improvement to the final results.

Table 5: Baseline comparison against winners of the LSMDC2016 movie clip retrieval challenge

<table>
<thead>
<tr>
<th>Method</th>
<th>R@1</th>
<th>R@5</th>
<th>R@10</th>
<th>Median Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yu et al. [42]</td>
<td>3.6%</td>
<td>14.7%</td>
<td>23.9%</td>
<td>50</td>
</tr>
<tr>
<td>Levi et al. [24]</td>
<td>4.7%</td>
<td>15.9%</td>
<td>23.4%</td>
<td>64</td>
</tr>
<tr>
<td>Our baseline</td>
<td>7.3%</td>
<td>19.2%</td>
<td>27.1%</td>
<td>52</td>
</tr>
</tbody>
</table>

Method R@1 R@5 R@10 Median Rank

5263
6. Conclusion

We have proposed an efficient online optimization method based on the Block-Coordinate Frank-Wolfe algorithm. We use this new algorithm to scale-up discriminative clustering model in the context of weakly supervised person and action recognition in feature-length movies. Moreover, we have proposed a novel way of handling the background class, which does not require collecting background class data as required by the previous approaches, and leads to better performance for person recognition. In summary, the proposed model significantly improves action recognition results on 66 feature-length movies. The significance of the technical contribution goes beyond the problem of person-action recognition as the proposed optimization algorithm can scale-up other problems recently tackled by discriminative clustering. Examples include: unsupervised learning from narrated instruction videos [1], text-to-video alignment [6], co-segmentation [16], co-localization in videos and images [18] or instance-level segmentation [32], which can be now scaled-up to an order of magnitude larger datasets.

Acknowledgments. This work has been supported by ERC grants ACTIVIA (no. 307574) and LEAP (no. 336845), CIFAR Learning in Machines & Brains program, ESIF, OP Research, development and education Project IMPACT No. CZ.02.1.01/0.0/0.0/15_003/0000468 and a Google Research Award.
References


[26] A. Miech, J. B. Alayrac, P. Bojanowski, I. Laptev, and J. Sivic. Supplementary material (appendix). https://hal.inria.fr/hal-01569540. 5, 8


