L1-regularized Logistic Regression Stacking and Transductive CRF Smoothing for Action Recognition in Video

Svebor Karaman
University of Florence
svebor.karaman@unifi.it

Lorenzo Seidenari
University of Florence
lorenzo.seidenari@unifi.it

Andrew D. Bagdanov
University of Florence
bagdanov@dsi.unifi.it

Alberto Del Bimbo
University of Florence
alberto.delbimbo@unifi.it

Abstract

For the 2013 THUMOS challenge we built a bag-of-features pipeline based on a variety of features extracted from both video and keyframe modalities. In addition to the quantized, hard-assigned features provided by the organizers, we extracted local HOG and Motion Boundary Histogram (MBH) descriptors aligned with dense trajectories in video to capture motion. We encode them as Fisher vectors. To represent action-specific scene context we compute local SIFT pyramids on grayscale (P-SIFT) and opponent color keyframes (P-OSIFT) extracted as the central frame of each clip. From all these features we built a bag-of-features pipeline using late classifier fusion to combine scores of individual classifier outputs. We further used two complementary techniques that improve on the basic baseline with late fusion. First, we improve accuracy by using L1-regularized logistic regression (L1LRS) for stacking classifier outputs. Second, we show how with a Conditional Random Field (CRF) we can perform transductive labeling of test samples to further improve classification performance. Using our features we improve on those provided by the contest organizers by 8%, and after incorporating L1LRS and the CRF by more than 11%, reaching a final classification accuracy of 85.7%.

1. Introduction

The THUMOS challenge [2] is part of the First International Workshop on Action Recognition with a Large Number of Classes. The objective is to address for the first time the task of large scale action recognition with 101 actions classes appearing in a total of 13,320 video clips extracted from YouTube [6]. These videos are structured in groups that correspond to a small set of videos where the same action is performed in the same environment.

The THUMOS organizers provided a set of five features: Motion Boundary Histograms (MBH), Histograms of Oriented Gradients (HOG), Histograms of Optical Flow (HOF), TR (Trajectories), and Space-Time Interest Points (STIP). These features are encoded within a Bag-of-Word pipeline according to a dictionary of 4000 words. The entire dataset was split between train and test samples three times, each split randomly selecting two-thirds of the data for training and the remaining data for testing. Videos from the same group never appear in both the training and test set. Our submission to the THUMOS competition consisted of four runs, which we briefly summarize here:

- **Run-1:** We used the features provided by the organizers and trained 1-vs-all SVMs with intersection kernels. Classifiers are trained independently on the five feature modalities and their outputs combined using late fusion by summing the classifier score of each feature for each sample.
- **Run-2:** We added our own scene descriptors, based on P-SIFT [5] and P-OSIFT (both local pyramidized versions of the SIFT and Opponent-SIFT descriptor), together with better quantized dense trajectory features: MBH, MBHx, MBHy and HOG.
- **Run-3:** We use L1-regularized logistic regression stacking (L1LRS) to better combine class/feature experts outputs from Run-2.
- **Run-4:** Finally we employ a Conditional Random Field (CRF) to enforce a correct labelling of similar clips using the outputs of the L1LRS combined outputs from Run-3.

The rest of the paper describes in more detail our experimental setup and the techniques used to generate the results from each of our submitted runs.

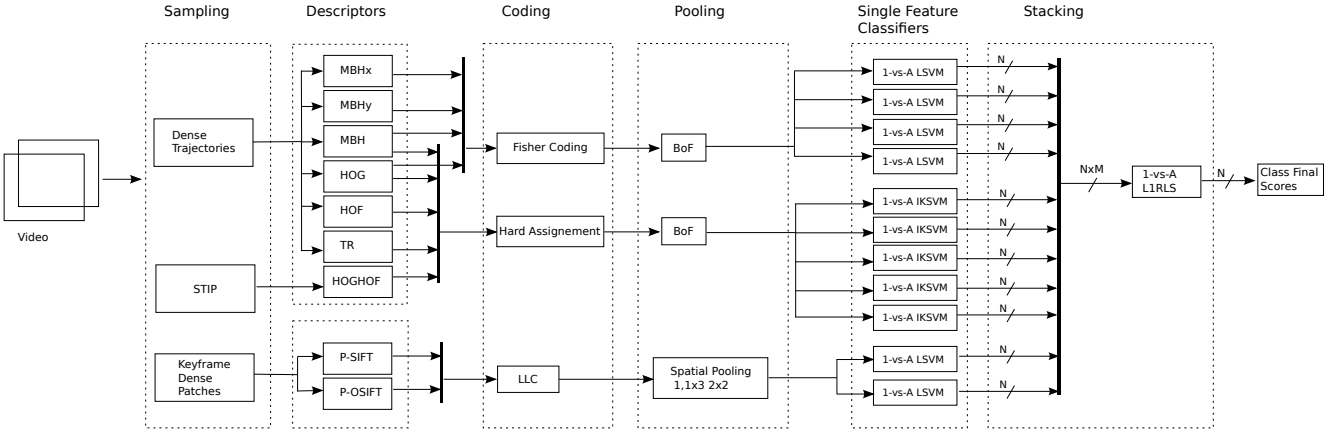

Figure 1: Our system pipeline. We exploit features sampled from STIP and Dense Trajectories on entire clips and Dense Patches on clip Keyframe. We employ HOG+HOF descriptors computed on STIP keypoints, and HOG, HOF, TR, MBH, MBHx and MBHy descriptors computed on dense trajectories. On keyframes extracted from each clip we compute P-SIFT and P-OSIFT on dense patches. Single feature, linear 1-vs-all classifiers are trained and the outputs are used as features for our L1-regularized logistic regression stacker.

2. Experimental setup

In this section we describe the experimental setup for our submission. We first give an overview of our pipeline in section 2.1, then describe the features used in section 2.2, and finally give details on the class experts and their parameters.

2.1. System Overview

Our approach focuses on late and "very late" fusion schemes. In figure 1 we give an overview of our pipeline. Each feature is coded, pooled and then used to learn a class expert classifier that relies on that single feature. The outputs are combined using a variety of late fusion techniques (described in detail in the description of each run below).

2.2. Features and feature encoding

In addition to the video features, we use P-SIFT and P-OSIFT on keyframes. The pyramidal SIFT (P-SIFT) and pyramidal Opponent-SIFT (P-OSIFT)[1] descriptor is constructed by varying the pooling resolution that controls the number and size of each subregion used to compute each histogram. A pyramidized descriptor consists of multiple SIFT descriptors that describe the patch at different levels of detail. Derivative scale is set according to the patch scale and number of pooling regions. We use three pooling levels of P-SIFT respectively, corresponding to 2×2, 4×4, 6×6 pooling regions.

P-SIFT and P-OSIFT descriptors are then coded using Locality-constrained Linear Coding (LLC) [7]. Each pooling configuration is coded with its own dictionary of sizes: 1500, 2500 and 3000, respectively. The final keyframe descriptor is obtained using a spatial pyramid with the following configuration: 1x1, 2x2, 1x3. For MBH and HOG we instead used the Fisher vector encoding [4] with 256 Gaussians after reducing the dimensionality of each descriptor to 64 using PCA. To enrich the set of available features, for the Motion Boundary Histogram features we include separate x- and y-components (MBHx and MBHy) as well as the standard concatenation of the two local descriptors (MBH).

[1]Source at: `http://www.micc.unifi.it/seidenari/projects/p-sift/`

2.3. Class/feature experts and parameter estimation

Our approach uses a set of experts, one for each feature/class pair. Assuming a set of M features $\mathcal{F} = \{f_1, f_2, \ldots, f_M\}$, and a classification problem over N classes $\mathcal{C} = \{c_1, c_2, \ldots, c_N\}$, we build a total of $N \times M$ experts. We denote by E_i^j the expert for class c_i that uses feature f_j. Each expert is a 1-vs-all SVM using either linear or histogram intersection kernel that maps from a vector $\mathbf{x}$ in the feature space of feature f_j to a decision value for membership of $\mathbf{x}$ in class c_i.

As features we include the five pre-quantized Bag-of-Words (BoW) histograms features provided the THUMOS organizers, plus the six new features described above. This gives us a total of 11 features, which along with the 101 classes in the challenge yields a total of 1,111 class/feature experts. For the BoW histogram features provided by the THUMOS organizers, we use histogram intersection kernels for SVM experts. The C value for of the BoW experts is obtained by cross validating for each given feature. For the features we computed on videos and keyframes, we use the Fisher vector encoding and linear SVMs with a C value fixed to 10. At times it will be convenient to refer the two

sets of features (the THUMOS features and ours) independently, and we decompose $\mathcal{F}$ as:

$$\begin{aligned} \mathcal{F} &= \mathcal{F}_{\text{org}} \cup \mathcal{F}_{\text{ours}} \text{ , where} \quad (1) \\ \mathcal{F}_{\text{org}} &= \{\text{MBH}_{\text{BoW}}, \text{HOG}_{\text{BoW}}, \text{HOF}_{\text{BoW}}, \text{TR}_{\text{BoW}}, \text{STIP}_{\text{BoW}}\} \\ \mathcal{F}_{\text{ours}} &= \{\text{P-SIFT}, \text{P-OSIFT}, \text{MBH}, \text{MBH}_x, \text{MBH}_y, \text{HOG}\} \end{aligned}$$

2.4. Summary of features and experts

For each feature (those provided by contest organizers, and those we extracted), we summarize here the encoding we use as well as the classifiers used as primitive single-feature experts for each class are:

- **MBH_{BoW}:** quantized MBH feature histograms of 4000 words provided by the organizers. Experts are 1-versus-all histogram intersection SVM classifiers.
- **HOG_{BoW}:** quantized HOG feature histograms of 4000 words provided by the organizers. Experts are 1-versus-all histogram intersection SVM classifiers.
- **HOF_{BoW}:** quantized HOF feature histograms of 4000 words provided by the organizers. Experts are 1-versus-all histogram intersection SVM classifiers.
- **TR_{BoW}:** quantized trajectory feature histograms of 4000 words provided by the organizers. Experts are 1-versus-all histogram intersection SVM classifiers.
- **STIP_{BoW}:** quantized STIP feature histograms of 4000 words provided by the organizers. Experts are 1-versus-all histogram intersection SVM classifiers.
- **P-SIFT:** computed on dense patches from keyframes, encoded using LLC over a three-level pyramid. Experts are 1-versus-all linear SVM classifiers.
- **P-OSIFT:** computed on dense patches from keyframes, encoded using LLC over a three-level pyramid. Experts are 1-versus-all linear SVM classifiers.
- **MBH:** concatenation of MBHx and MBHy computed from dense trajectories from entire clips, encoded using Fisher vectors with 256 Gaussians. Experts are 1-versus-all linear SVM classifiers.
- **MBH_x:** the x-component of the Motion Boundary Histogram computed on dense trajectories from entire clips, encoded using Fisher vectors with 256 Gaussians. Experts are 1-versus-all linear SVM classifiers.
- **MBH_y:** the y-component of the Motion Boundary Histogram computed on dense trajectories from entire clips, encoded using Fisher vectors with 256 Gaussians. Experts are 1-versus-all linear SVM classifiers.
- **HOG:** computed on dense trajectories from entire clips, encoded using Fisher vectors with 256 Gaussians. Experts are 1-versus-all linear SVM classifiers.

3. Experiments

Here we detail each of the four runs we submitted to the THUMOS 2013 challenge.

3.1. Run-1: baseline performance

For our first run we used only the class/feature experts derived from the features provided by the organizers (i.e. STIP, MBH, TR, HOG, HOF). We trained single feature classifiers in a 1-vs-all setting. Clip $\mathbf{x}$ is classified as belonging to class c using late fusion of all single-feature class experts. All scores of single feature classifiers are summed and the class is selected as following:

$$\text{class}(\mathbf{x}) = \arg\max_c \sum_{f \in \mathcal{F}_{\text{org}}} E_c^f(\mathbf{x}) \quad (2)$$

This is our basic fusion rule, but note that the sum is limited to features in $\mathcal{F}_{\text{org}}$. With this setting, using only the quantized feature histograms provided by the contest organizers, we obtain 74.6% classification accuracy and 0.760 mAP.

3.2. Run-2: Improved motion features and contextual information

To improve accuracy we added two sets of features extracted from video clips and keyframes. The idea behind this is that the LLC-coding of our P-SIFT and P-OSIFT features sampled densely on a single keyframe for each clip should provide a sort of contextual information about each action class. The addition of Fisher vector encodings of MBH_x, MBH_y, MBH and HOG features should additionally allows us to capture richer representations than the 4000 codeword, pre-quantized features used in Run-1. The experts for all of these new features are linear SVM classifiers. Note that these features are used *in addition* to those already used in Run-1. The late fusion of unweighted classifier scores is again performed as in equation (2), but incorporating all feature experts:

$$\text{class}(\mathbf{x}) = \arg\max_c \sum_{f \in \mathcal{F}} E_c^f(\mathbf{x}) \quad (3)$$

Using all available features we achieve 82.4% accuracy with Run-2, an improvement of 7.8% over the Run-1 baseline, and 0.836 in mAP.

3.3. Run-3: stacking classifiers with L1LR

For our next run we implemented a more sophisticated fusion technique based on stacking of classifier outputs. We compute a new feature from the concatenation of all the single feature classifier scores. We then train an L1-regularized logistic regression on these features so that the regression predicts the class and the enforced sparsity performs feature selection.

For a clip $\mathbf{x}$, we define the stacked representation as the concatenation of all class/feature experts evaluated on it:

$$S(\mathbf{x}) = [E_1^1(\mathbf{x}), E_1^2(\mathbf{x}), \ldots, E_2^1(\mathbf{x}), \ldots, E_N^M(\mathbf{x})] \quad (4)$$

$$= [E_i^j], \text{ for } j \in \{1, \ldots M\},\ i \in \{1, \ldots N\} \quad (5)$$

Recall that each element of $E_i^j(\mathbf{x})$ is an unweighted, SVM output from a class/feature expert.

In order to robustly estimate a logistic regression model that does not overfit the training set, expert scores of training samples must be computed on held out data. We thus split the training set into five independent folds, all respecting the groups provided in the THUMOS data (i.e. each group appears exactly once in a held-out part). We compute decision values for all 1-vs-rest SVM experts trained on each feature and build a new feature representation of each clip by concatenating the out-of-sample classifier scores on the training set as described above. We then train L1-regularized logistic regression models for stacking (L1LRS) on this representation. The logistic regression model for each class $c \in \mathcal{C}$ minimizes the loss:

$$(\boldsymbol{\beta}_c, b_c) = \arg\min_{\beta, b} ||\boldsymbol{\beta}||_1 + C \sum_{i=1}^{n} \ln(1 + e^{-y_i \boldsymbol{\beta}^T S(\mathbf{x}_i) + b}) \quad (6)$$

where $\mathbf{x}_i$ is a feature space representation of clip i, and y_i is its corresponding class label. The new expert for class c is the linear predictor defined by the projection $\boldsymbol{\beta}_c$ of the stacked original experts $S(\mathbf{x})$, along with the bias term b_c.

By sparsely combining the outputs of all experts, the L1LRS method achieves 84.4% average accuracy, an improvement of 2% over Run-2, and 0.849 mAP. It is illuminating to study in depth the usage of class/feature experts by the logistic regression models broken down independently by class and feature. To characterize the difficulty of a class we consider the following retrieval problem over individual class experts. Considering a clip $\mathbf{x}$ of class c, for M features and N classes we have $N \times M$ scores in the stacked representation $S(\mathbf{x})$ from the individual class/feature experts. A clip $\mathbf{x}$ of class c is considered "easier" if all its class experts, that is the experts for class c trained on the M features, have a high score with respect to experts from other classes. Otherwise, a clip is considered "harder" if the correct class experts are not adequate for discriminating it and non-class experts are thus needed in the final regression model. To quantify these concepts, we consider the problem of "retrieving" class experts using a clip as query. For a given clip $\mathbf{x}$, we rank all experts according to their score $E_i^j(\mathbf{x})$. We then consider experts corresponding to the correct class of clip $\mathbf{x}$ as relevant, and all others not relevant. "Easier" clips will have higher AP with respect to "harder" ones. The average precision of each clip then measures its "hardness", and the mAP is used to quantify the difficulty of each class

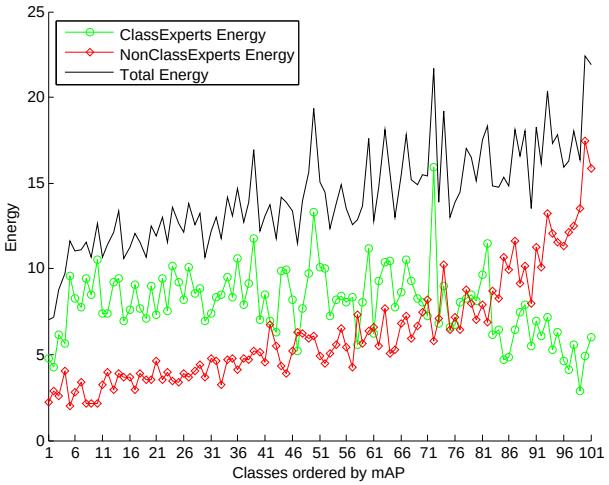

Figure 2: Energy of the coefficients in the final model for class experts and non class experts with respect to decreasing expert mAP-ordered classes.

in terms of class experts. Therefore "harder" classes will have a lower mAP. We refer to this as *expert mAP*.

In figure 2 we illustrate the coefficient energy $||\boldsymbol{\beta}_c||^2$ of each L1LRS model ordered by decreasing expert mAP. In the figure we plot the total energy, the expert energy (only experts of the correct class), and non-expert energy. The coefficient energy is a measure of the density of the learned coefficients $\boldsymbol{\beta}_c$, and a sparse vector will have a lower energy than a denser one. Observe the general trend of increasing energy (thus decreasing sparseness) of all model coefficients, showing that for more difficult classes the L1LRS tends to yield less sparse solutions than for easier ones. This plot also shows that for the easier classes the L1LRS gives more energy to the weights of class experts while the hardest classes will have higher energy for non class-expert weights. When class experts are not reliable enough, the L1LRS relies on other class experts for negative support.

To better understand the usage of feature-specific class experts by the L1LRS models, we break down the weights of each feature for each class in figure 3. In this figure the contribution of each feature is indicated as a percentage of the total contribution of all features as measured by coefficient energy each $\boldsymbol{\beta}_c$. We can see that for most of the classes (exactly 96 out of 101), our features are given a higher proportion of the total coefficient energy in the stacked classifier. The dominance of reddish colors indicates the importance of the MBH, MBH_x and MBH_y features. These features are powerful for characterizing human actions as they describe motion boundaries.

Some actions also benefit from the addition of the contextual scene descriptors P-SIFT and P-OSIFT. The four classes using these descriptors the most are: 'BreastStroke', 'FieldHockeyPenalty', 'BaseballPitch' and 'PlayingPiano',

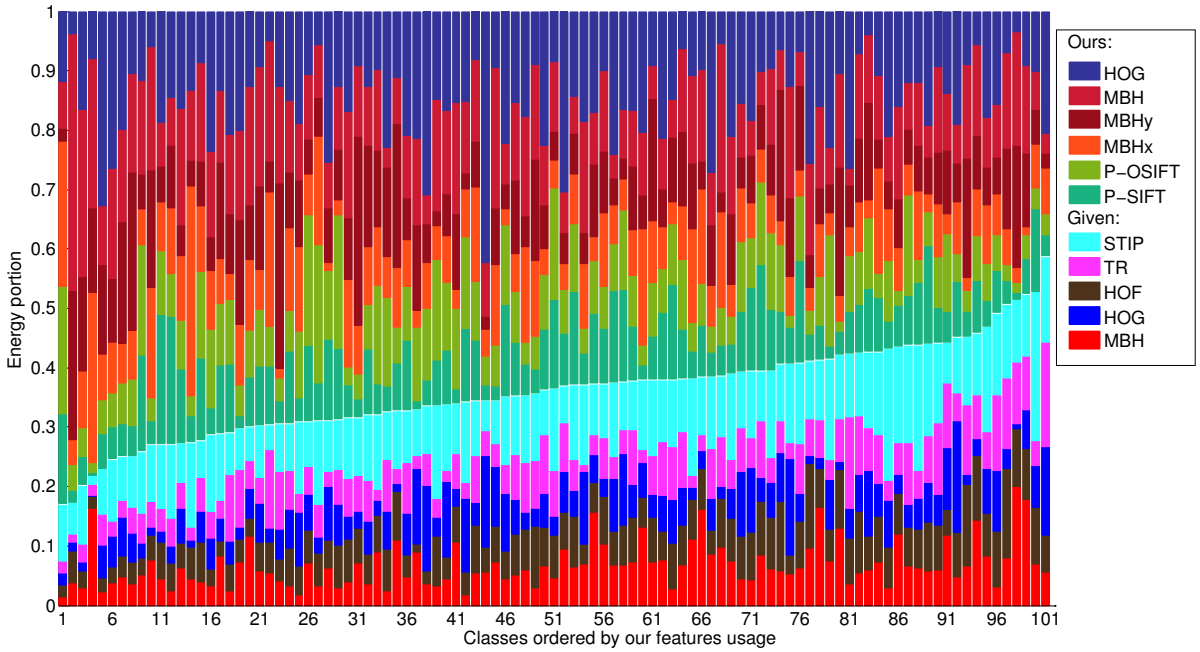

Figure 3: Feature usage as coefficient energy proportion on classes ordered by decreasing usage of our additional features.

which are respectively at positions 1, 9, 29 and 51 in figure 3. In figure 4 we illustrate the mean keyframe from these classes. These classes are all characterized by a specific and distinctive environment that appear as a clear and constant background (swimming pool, playfield) or persistently present object identifiable even in the mean keyframe (a piano). On the contrary, classes that use the least the contextual features are actions where the humans are fully framed and with higher motion. Indeed, these classes tend to rely heavily on motion features. For example, the L1LRS model of class 'PommelHorse' have 84.4% of its energy dedicated to the 4 different types of MBH features.

3.4. Run-4: transductive labeling CRFs

For our final run we applied a CRF to perform transductive smoothing of assignments of class labels to test video clips. The CRF uses the L1LRS output as unary potentials, and distances between stacked expert outputs are used to build a k-NN graph defining the CRF topology. The smoothness potential uses the similarity between stacked features to force similar samples to take the same label. This idea of transductive labeling was recently applied to re-identification in [3]. The CRF is defined as a graph $G = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of nodes corresponding to all samples (both training and test) and $\mathcal{E}$ is the set of edges connecting each sample to its k nearest neighbors.

Formally, we aim at finding a labeling $\hat{\mathbf{c}} = [c_i]$ associating labels to all clips ($c_i \in \mathcal{C}$ is the label associated with node $v_i \in \mathcal{V}$ by $\hat{\mathbf{c}}$) and that minimizes the energy:

$$W(\hat{\mathbf{c}}) = \sum_{i \in \mathcal{V}} \phi_i(\hat{c}_i) + \lambda \sum_{(v_i, v_j) \in \mathcal{E}} \psi_{ij}(\hat{c}_i, \hat{c}_j), \tag{7}$$

where λ is a tradeoff factor controlling the relative contribution of unary and binary costs. The data cost of a test sample $\mathbf{x}_i$ labeled as $\hat{c}_i$ is defined as:

$$\phi_i(\hat{c}_i) = e^{-\left(\boldsymbol{\beta}_{\hat{c}_i}^T S(\mathbf{x}_i) + b_{\hat{c}_i}\right)}, \tag{8}$$

where $S(\mathbf{x})$ are the stacked expert outputs on clip $\mathbf{x}$, and $(\boldsymbol{\beta}_{\hat{c}_i}, b_{\hat{c}_i})$ is the L1LRS model for class $\hat{c}_i$. The data cost of training samples is manually set to 0 for the correct class and 1 for all incorrect classes.

All training samples of the same class are connected in the topology, and each test sample is connected to its k-nearest neighbors from both training set and test sets (we use $k = 12$). The smoothness potential is:

$$\psi_{ij}(\hat{c}_i, \hat{c}_j) = \psi_{ij}\psi(\hat{c}_i, \hat{c}_j). \tag{9}$$

The edge weight ψ_{ij} encodes the similarity of $\mathbf{x}_i$ and $\mathbf{x}_j$ in terms of *expert agreement*:

$$\psi_{ij} = \exp\left(-\frac{||S(\mathbf{x}_i) - S(\mathbf{x}_j)||_2}{\sigma_i \sigma_j}\right), \tag{10}$$

where σ_i and σ_j are local scaling [8] factors estimated as the distance between the sample $\mathbf{x}_i$ and its $2k$-nearest neighbors. The factor ψ_{ij} is maximized when all experts are in agreement on the two samples.

Figure 4: Mean of keyframes of the classes that use contextual features the most: BreastStroke, BaseballPitch, FieldHockeyPenalty, PlayingPiano; and the least: PommelHorse, CleanAndJerk, JumpRope, BodyWeightSquats.

The label consistency $\psi(\hat{c}_i, \hat{c}_j)$ encodes the intrinsic similarity between class $\hat{c}_i$ and $\hat{c}_j$. It is estimated on the training set as the sum of the outputs of all classifiers of $\hat{c}_i$ on training samples of $\hat{c}_j$ and of the output of all classifiers of $\hat{c}_j$ on training samples of $\hat{c}_i$. Using C_i and C_j to represent training samples from class i and class j, respectively, the smoothness cost is:

$$\psi(\hat{c}_i, \hat{c}_j) = \exp\left(-\frac{D(\hat{c}_i, \hat{c}_j) + D(\hat{c}_j, \hat{c}_i)}{2}\right), \qquad (11)$$

where $D(\hat{c}_i, \hat{c}_j)$ is an average measure of class expert $\hat{c}_j$ acceptance of training samples from class $\hat{c}_i$:

$$D(\hat{c}_i, \hat{c}_j) = \frac{1}{|\mathcal{F}||C_i|} \sum_{f \in \mathcal{F}} \sum_{\mathbf{x} \in C_i} E_j^f(\mathbf{x}). \qquad (12)$$

After constructing the CRF with these unary and binary cost functions, we then use graph-cuts [1] to find the optimal labeling of vertices corresponding to test images. This labeling procedure enforces local consistency of the labels. For Run-4 we achieve an accuracy of 85.7% which is an improvement of 1.3% in accuracy over Run-3 and of 11.1% over the Run-1 baseline. Since we use graph cuts to solve the labeling problem, we cannot compute mAP as we do not have a ranking of labels at each vertex in the CRF.

4. Discussion

Table 1 summarizes the results for each of our four runs on the THUMOS challenge. In addition to the features provided by the THUMOS organizers, we added a representation for the context of the action based local, pyramidal SIFT and Opponent SIFT descriptors computed on a keyframes of clips. In our submission we also investigated the use of a more powerful encodings of local features, for example Fisher vectors and LLC. Our runs combine all available features from video and keyframes in a late fusion framework. Our results demonstrate the complementarity of these features.

	$\mathcal{F}_{\text{org}}$	$\mathcal{F}_{\text{ours}}$	L1LRS	CRF	Accuracy
Run-1	✓				74.6%
Run-2	✓	✓			82.4%
Run-3	✓	✓	✓		84.4%
Run-4	✓	✓	✓	✓	**85.7**%

Table 1: Summary of our four runs.

We have also studied two "very late" fusion schemes. We used stacking of class/feature experts with L1-regularized logistic regression and showed of how it works to automatically adjust the weights of different experts. Weights learned by the L1-regularized classifier are strongly related to the difficulty for the action to be recognized. Finally, we used a CRF to perform transductive labeling relying on both the smarter L1LRS classifier outputs but also on nearest neighbors in the feature space of stacked experts, which further improved classification performance.

References

[1] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[2] Y.-G. Jiang, J. Liu, A. Roshan Zamir, I. Laptev, M. Piccardi, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. `http://crcv.ucf.edu/ICCV13-Action-Workshop/`, 2013.

[3] S. Karaman and A. D. Bagdanov. Identity inference: generalizing person re-identification scenarios. In *Proceedings of ECCV - Workshops and Demonstrations*, pages 443–452, 2012.

[4] F. Perronnin, J. Sanchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *Proc. of ECCV*, 2010.

[5] L. Seidenari, G. Serra, A. D. Badanov, and A. Del Bimbo. Local pyramidal descriptors for image recognition. *Transactions on Pattern Analisys and Machine Intelligence*, in press, 2013.

[6] K. Soomro, A. Roshan Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. In *CRCV-TR-12-01*, 2012.

[7] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *Proc. of CVPR*, 2010.

[8] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *Advances in neural information processing systems*, pages 1601–1608, 2004.