Incorporating Temporal Context in Bag-of-Words Models

Tamar Glaser
Technion
Haifa, Israel
tamarglasr@gmail.com

Lihi Zelnik-Manor
Technion
Haifa, Israel
lihi@ee.technion.ac.il

Abstract

Bag-of-Words (BoW) is a highly popular model for recognition, due to its robustness and simplicity. Its modeling capabilities, however, are somewhat limited since it discards the spatial and temporal order of the codewords. In this paper we propose a new model: Contextual Sequence of Words (CSoW) which incorporates temporal order into the BoW model for video representation. The temporal context is incorporated in three scales that capture different aspects of the variability between different performances of the same action. We show that using CSoW instead of BoW leads to a significant improvement in action recognition rates, on several different setups.

1. Introduction

A highly popular approach for action recognition is based on the Bag-of-Words (**BoW**) model [2, 8, 10, 20]. Its popularity is due to simplicity and robustness hand-in-hand with state-of-the-art results. In BoW one first collects a large sample of feature descriptors from a corpus of training videos. A dictionary is then constructed by quantizing the space of feature descriptors into code-book "words". Finally, a video is represented by the corresponding histogram of code-book words. This approach was shown to provide impressive results in action recognition [3, 8, 10, 20]. Its main drawback stems from discarding all the temporal and spatial relations between the features in the video, i.e., ignoring the context.

Temporal context information is essential for action representation for three reasons. First, many actions are composed of the same sub-actions albeit at a different order. For instance, both hurdle-race and long-jump are composed of running and jumping, however, the order of these sub-actions is different. In the standard BoW approach such actions could be easily confused as one since the model ignores the order of events.

Second, when different people perform the same activity, they often vary in the time spent on each sub-action. For example, when chopping a banana, one person could spend a longer time peeling it slowly while another could linger chopping it carefully. In the BoW model these two performances could be mistaken for two different actions as the corresponding feature statistics are different.

Last, the quantization of the descriptor space into codewords introduces further errors as similar sub-actions could be erroneously grouped as one. For example, both punching and drinking start from raising the arm. Such sub-actions could be differentiated if one were to consider also the context in which the sub-actions occur. In drinking the arm is raised and then bent, while in punching it is first raised and then stretched.

Previous attempts have been made for incorporating temporal context. Correlograms [12, 17] capture the location relations between features, at the price of ignoring the features' appearance. N-grams [6] extend BoW by considering sequences of N code-words as features. This enlarges the dictionary size exponentially, hence, in practice $N \leq 3$, i.e., only instantaneous context is captured. Finally, several other approaches which capture temporal order have been proposed, e.g., [13], however these do not relate directly to the BoW model. Our interest focuses on extending the BoW model due to its high popularity. In this paper we deal with incorporating the temporal context alone, however, this could be easily extended to consider also the spatial context.

In this work we incorporate temporal relations by representing videos using a *Contextual Sequence of Words* (**CSoW**). A CSoW is a sequence of codewords which preserves the temporal context within the sequence in three scales. The coarse scale captures variations between the global order of sub-actions. The medium scale provides robustness against differences in speed between different performances of the same action. Finally, the fine scale differentiates between similar sub-actions by considering the temporal context in which they occur. We further present metrics by which the distance between two contextual sequences of words is measured. Our experiments show that CSoW leads to a significant improvement in action recognition compared to BoW.

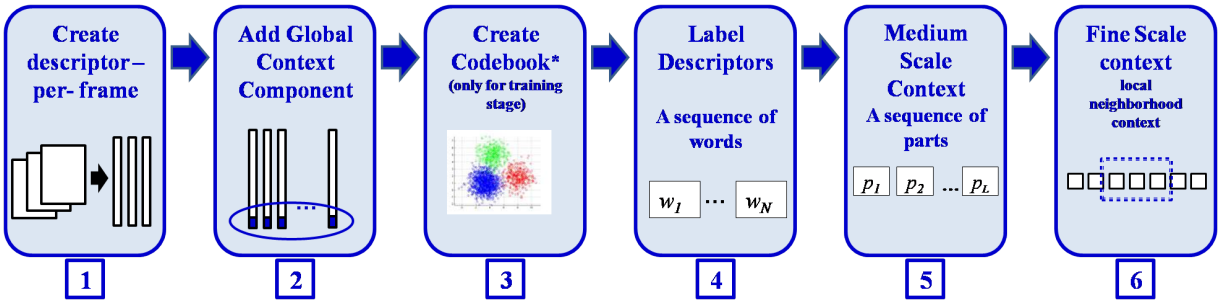


Figure 1. The CSoW representation scheme.

The rest of the paper is organized as follows. Our overall framework is presented in section 2 and the details of our approach to incorporating global, medium and fine scale temporal context are provided in section 3. In section 4 we describe how the distance between two videos is measured. We present quantitative empirical evaluation in section 6 and conclude in section 7.

2. Framework

In this section we define the Contextual Sequence of Words (**CSoW**) representation. CSoW is similar to BoW, in the sense that it uses a code-book, and labels the descriptors using codewords. It differs in that rather than using a histogram over the codewords we also consider their order of appearance in the video. CSoW encodes the temporal context of a video sequence in three scales: global, medium and fine scale. For clear exposition we assume for the time being a single descriptor per frame. Later on in section 5 we will relax this assumption and show that our CSoW can be applied when a descriptor is extracted for each interest point.

As illustrated in Figure 1 our framework consists of the following steps:

1. Extract a descriptor for each frame in the video.
2. ***Add Global Context***, which indicates the relative temporal location of each descriptor along the sequence. This stage is inspired by [11], where global spatial position was used (section 3.1).
3. ***Codebook Generation***: In the training phase we generate a codebook of V codewords using K-means clustering algorithm. Let N_{cls} be the number of classes. We generate N_{cls} codebooks of $\frac{V}{N_{cls}}$ words by clustering the descriptors of each action class separately, and then all N_{cls} codebooks are gathered into one 'merged' codebook. Since our descriptors already incorporate global context, so does the codebook.
4. ***Labeling:*** Each frame is labeled according to its nearest codeword. After this stage a video sequence is represented as a sequence of "words".
5. ***Medium scale context:*** Different occurrences of the same action could vary temporarily in the length of the sub-actions. Therefore, we aggregate the "words" into "parts" which correspond to the sub-actions, as described in section 3.2.
6. ***Fine scale context:*** Differentiating between sub-actions can be improved by using the internal order of sub-actions in their local neighborhood. We incorporate this fine-scale context information as described in section 3.3.

Code-book creation is done obviously only during training. A given test video sequence, is represented directly using global, medium and fine scale, by labeling the extended descriptors to codewords from a precomputed code-book.

3. Incorporating Temporal Context

Let $I_1, \ldots, I_N$ be N frames of a video. We start by extracting a feature vector f_i, describing each frame I_i. After this step the video is represented as:

$$video = [f_1, f_2, \ldots, f_N]. \tag{1}$$

3.1. Global Context

In many cases, different actions are composed of similar components. For instance, a security camera observing an entrance door sees "Enter" as: walk – open door – walk – close door. Similarly, "Exit" is composed of: open door – walk – close door – walk. Both entering and exiting actions are composed of similar sub-actions while they differ in the sub-actions order. Descriptors extracted from the walking at the beginning of "Enter" will most likely be assigned to the same codeword as descriptors from the walking at the end of "Exit". This will make differentiating between the two actions more difficult.

To avoid such mix-ups we wish to discriminate between sub-actions occurring at different moments along the action. This can be achieved by incorporating the relative moment of occurrence into the feature descriptors, as follows:

$$\bar{f}_i = \begin{bmatrix} f_i \\ \frac{i}{N} \end{bmatrix}. \tag{2}$$

In training stage we learn a dictionary by quantizing the extended features. Then, each feature $\bar{f}_i$ (both in training and test set) is labeled according to the closest dictionary word $w\{\bar{f}_i\}$.

After the labeling process a video is represented by a sequence of codewords which replace the extended features:

$$video = [w\{\bar{f}_1\}, w\{\bar{f}_2\}, \ldots, w\{\bar{f}_N\}]. \quad (3)$$

3.2. Medium Scale Context

Typically, different people perform the same activity at a somewhat different pace. For example, when chopping a banana one person would take longer to peel the banana, while the other could spend more time slicing it carefully. As illustrated in Fig. 3.(b) the code-word representations of such sequences share the same words, while they differ in the number of occurrences of each word (the length of each sub-action). To obtain robustness to this diversity we wish to capture the order in which the sub-actions occur while ignoring their individual length.

Inspired by [16] we achieve this by clustering the labeled descriptors according to their temporal location. Consecutive or close repetitions (up to 1/6 of the video length) of the same code-word are aggregated into a single sub-action or *"part"*, as illustrated in Figures 2, 3. As can be seen, after clustering the words into parts the similarity between the sequences is more evident.

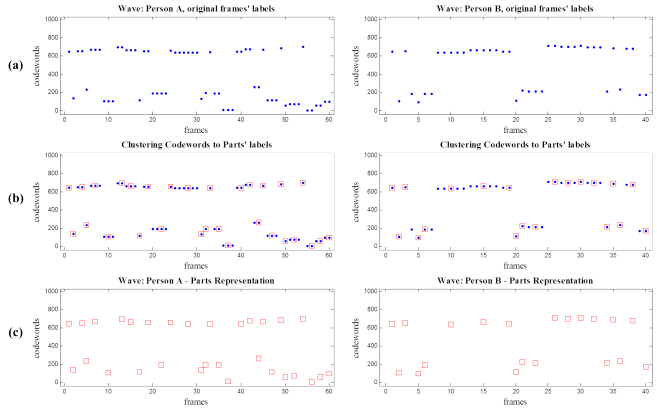


Figure 2. (a): Codewords vs. frames for two different sequences of the same action - "wave". (b) Clustering of codewords to parts. (c) Parts vs. frames for the same sequences. The similarity between the representations is more evident now.

Formally, after this step a video is represented as a sequence of L *parts*:

$$video = [p_1, \ldots, p_L], \quad (4)$$

where

$$p_i = \{w_i, loc_i, context_i\} \quad (5)$$

are the parts, that are characterized by their codeword w_i, the mean temporal location $loc_i = mean\{loc\,(\text{frames labeled to } w_i \text{ and clustered to } p_i)\}$, and their fine scale context $context_i$, to be described in the following section.

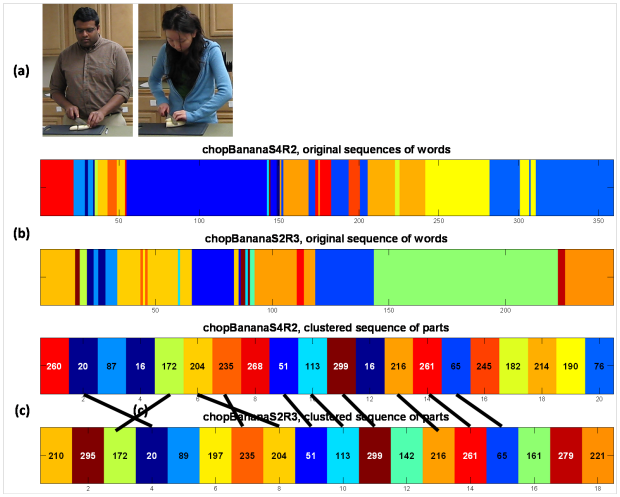


Figure 3. Action "chop banana" performed by two different actors. (a) Key frames from the original videos. (b) The corresponding labeled sequences (2 top rows), where each frame's descriptor is labeled by a codeword. (c) The corresponding sequences of parts, (2 bottom rows) after the medium scale context stage - the similarity between the videos is more evident.

3.3. Fine Scale Context

Often different actions include the same sub-actions (parts), however, the parts differ in the local context in which they are embedded. To capture this, we extend the parts representation to include their local neighborhood temporal context. In the following, we suggest and experiment with two options: one which uses the parts order explicitly, and one which encodes the contextual information of the local neighborhood in a more loose manner.

3.3.1 Rigid Fine Scale:

The rigid approach captures the order of parts in a local neighborhood of each part. We represent each part as $p_i = p_i^{rgd} = \{w_i, loc_i, context_i^{rgd}\}$, where

$$context_i^{rgd} = [w(p_{i-R}), \ldots, w(p_i), \ldots, w(p_{i+R})] \quad (6)$$

and R is the local neighborhood size.

3.3.2 Loose Fine Scale:

In the loose approach, we want to consider the local neighborhood, yet represent it in a more robust manner. Inspired by [16], we capture local context by histogramming the codewords in each part's local neighborhood, as follows.

A part p_i is represented by a pair of histograms, one over the codewords proceeding it:

$$h_i^{pre} = hist\left([w_{i-R}, \ldots, w_{i+lp}]\right),$$

and the other over the codewords following it:

$$h_i^{post} = hist\left([w_{i-lp}, \ldots, w_{i+R}]\right),$$

where lp is a small overlap between the histograms, and R is the local neighborhood size.

The local context in the case of loose fine scale is:

$$context_i^{los} = \left[h_i^{pre}, h_i^{post}\right]. \quad (7)$$

Finally, we represent each part as

$$p_i = p_i^{los} = \{w_i, loc_i, context_i^{los}\}.$$

Note that the histograms are performed over the original *codewords* in the parts' local neighborhood, and not the *parts* in the local neighborhood.

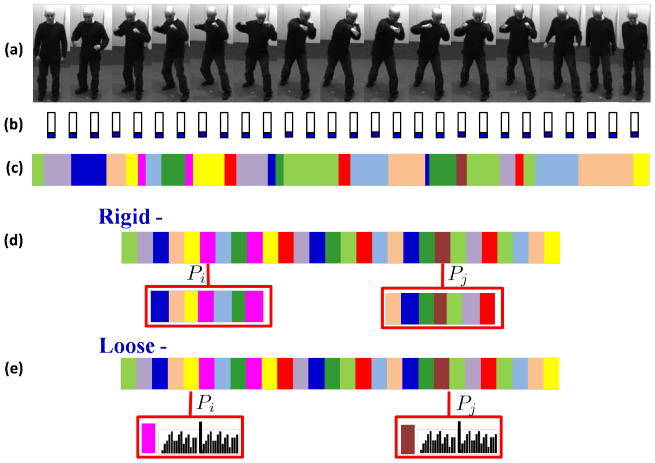


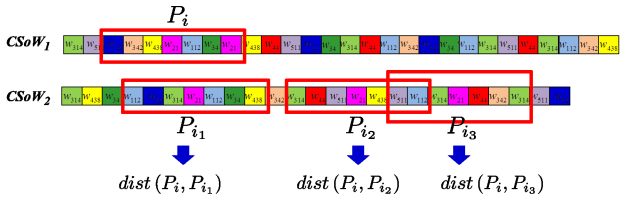
Figure 4. CSoW Representation Summary: (a) Video sequence frames (b) Extracting descriptors and adding global temporal context component. (c) Quantized descriptors - sequence of words (d) Rigid CSoW representation: Sequence of parts (clustered words). Each part is represented also by its local context - the sub sequence of parts in its local neighborhood. (e) Loose CSoW representation - each part is represented by the histogram of words in its preceding and following local neighborhoods.

Figure 4 summarizes the CSoW representation process.

Local Neighborhood Size When referring to local neighborhood, a definition of the neighborhood size is required. In the case of rigid fine scale context, the local neighborhood size radius is the number of preceding/following parts which are counted in the local neighborhood. In the case of loose fine scale context - it is the number of codewords from which the preceding/following histograms are built. Typically, the neighborhood size R required for the loose context is larger than for the rigid context since histograms are based on statistics. We report the values used for the local neighborhood in section 6, and discuss its affect in Section 6.3. In practice, the two methods show robustness to the neighborhood size R.

4. CSoW Representation Distance Definition

In order to complete the representation, a definition for the distance between a pair of CSoWs is required. The robustness to action's length and synchronization is one of the strengths of BoW model, and we would like to preserve this advantage.



$$dist\,(P_i) = \min_j \{dist\,(P_i, P_{ij})\}$$

$$dist = avg\,(dist(P_i))$$

Figure 5. Medium scale context distance computation between two CSoWs: parts matching. Parts $P_{i_1}, P_{i_2}, P_{i_3}$ in $CSoW_2$ are found as potential matches to part P_i in $CSoW_1$. All 3 distances to part P_i are computed, and the minimal one is chosen. The process repeats for all parts of $CSoW_1$, and the distance between the pair of CSoWs is the average of the computed distances for all parts (in one of the CSoWs).

To define the distance between two CSoWs we first find matches between parts across the CSoWs. The distance between two CSow's is taken as the average distance between matched parts. This is defined next.

Let $P = [p_1, \ldots, p_{L_1}]$ and $Q = [q_1, \ldots, q_{L_2}]$ be two CSoWs, with $L_1 \geq L_2$. The distance between P, Q is as follows:

1. For each part p_i in P we find a set of candidate matches Q_{p_i} in Q as parts with an identical or similar codeword, i.e., $q_j \in Q_{p_i}$ if:

$$dist\,(w(p_i), w(q_j)) < threshold. \quad (8)$$

2. When $Q_{p_i} \neq \emptyset$ we find a match for part p_i out of the candidate matches according to the minimal temporal context distance:

$$q_j = \underset{q_j \in Q_{p_i}}{\operatorname{argmin}} \{dist\,(context(p_i), context(q_j))\}. \quad (9)$$

We define the distance between contexts in section 4.1.

3. After finding the match to part p_i the distance for this part is computed:

$$dist(p_i) \triangleq$$
$$dist\,(p_i, q_j) = dist\,(context(p_i), context(q_j)) \quad (10)$$

4. Finally, the distance between P and Q is the average distance over all matches:

$$dist\,(P, Q) = \underset{1 \leq i \leq L_1}{\text{Mean}} \{dist\,(p_i, q_j)\}. \quad (11)$$

For parts with no match - $Q_{p_i} = \emptyset$ - we penalize with a "no-match" distance:

$$dist\left(p_i\right) = \left(2R+1\right) \max_{l,k} \left(dist\left(w_l, w_k\right)\right), \quad (12)$$

where R is the local neighborhood size.

We allow multiple matches for each part in Q since often two CSoWs of the same action have different lengths. This also considers the case of periodic actions. Note, that the multiple match enabling is limited by the global context component. Similar frames in distant temporal locations along an action can become similar parts only if they are similar enough relative to the temporal distance between them. Therefore, there is always a tradeoff between content similarity and temporal location similarity, when comparing descriptors.

The use of positive $threshold$, thus allowing matching parts to have similar codewords instead of requiring identical codeword, enables robustness to one of the model's main parameters - the codebook size (this is also BoW model main parameter). Since there is never certainty regarding the exact suitable codebook size, the request for similarity instead of identity in parts' codewords, enables some compensation in case of too large codebooks. It also increases the robustness to variability in action's performance, which may result in similar codewords and not identical ones. We set $threshold = 0.0015$.

4.1. Distance between Action Parts

Next, we define the distance between parts according to one of two fine scale context representation options, rigid or loose.

Distance between "Rigid" Parts: Comparing between a pair of sub-sequences while strictly enforcing the interior order within them is somewhat similar to alphabetical strings comparison. Therefore, we propose an extension of *Damerau-Levenshtein edit distance* [4] which is commonly used as a distance between alphabetic strings.
The original Damerau-Levenshtein edit distance counts the number of edit operations which are required in order to transform one string to the other. Edit operations are: Insertion / deletion of a single character, substituting a single character with another character, and transposing the locations of two adjacent characters. Edit distance therefore considers the order of the compared sequences.

Our local sub-sequences are not exactly strings of alphabetic characters, but codewords. In typos, which is one of the common usages of edit-distance, we would probably like to 'charge' a higher cost for replacing a 'f' with an 'a', than for replacing an 'e' with an 'a'. Similarly, when the characters are codewords (parts' labels), we would like to consider the distances between codewords when computing the distance between sub-sequences.

Our extension for Damerau-Levenshtein edit distance extends the original distance by using weighted costs for edit operations (denoted by WDL). The edit operations costs are relative to the distance between the codewords participating in the edit operation. The different edit operations costs are detailed in Eq. (13-18).

$$dist^{rgd}\left(context^{rgd}(p_i), context^{rgd}(q_j)\right) = WDL\left(context^{rgd}(p_i), context^{rgd}(q_j)\right) \quad (13)$$

where the cost of operations are: The cost of substitution/transposition between two codewords w_1, w_2 is the distance between the codewords:

$$substitution\left(w_1, w_2\right) = dist\left(w_1, w_2\right), \quad (14)$$

$$transposition\left(w_1, w_2\right) = dist\left(w_1, w_2\right). \quad (15)$$

The cost of codeword insertion/deletion is:

$$insertion = \frac{1}{2}\left(dist\left(w_{insert}, w_{precede}\right) + MX\right), \quad (16)$$

$$deletion = \frac{1}{2}\left(dist\left(w_{del}, w_{follow}\right) + MX\right). \quad (17)$$

where MX is the maximal cost, which is the maximal distance between codewords:

$$MX = \max_{l,k}\left(dist\left(w_l, w_k\right)\right). \quad (18)$$

Distance between "Loose" Parts In the loose case the fine scale context representation is a pair of histograms:

$$context_i^{los} = \left[h_i^{pre}, h_i^{post}\right].$$

Thus the distance is the sum of the distances between the histograms. The distances between histograms is computed using χ^2 distance:

$$dist^{los}\left(context^{los}(p_i), context^{los}(q_j)\right) = \chi^2\left(h_{p_i}^{pre}, h_{q_j}^{pre}\right) + \chi^2\left(h_{p_i}^{post}, h_{q_j}^{post}\right). \quad (19)$$

This maintains the idea of strengthening the robustness over enforcing interior order, and allowing more variability within the sub-actions.

5. Descriptor-per-Interest-Point Extension

For simplicity, CSoW was presented in the previous sections as a descriptor-per-frame video representation. The extension to a descriptor-per-interest-point (e.g., [9]) is trivial. We first extract features at each interest point and add

global context. Interest points detected on the same frame are assigned the same global context component value. The feature vectors are then quantized into codewords. Next, we group code-words into parts, as described before. Since the temporal location of each part is the mean location of words assigned to it, this step yields a temporally-ordered sequence of parts $[p_1, \ldots, p_L]$, as in the descriptor-per-frame case. Fine scale temporal context remains unchanged.

6. Experimental Evaluation

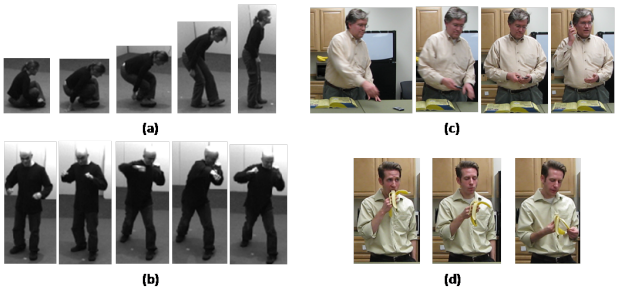


Figure 6. Sample frames from the tested datasets. (a) "get-up" and (b) "punch" actions from IXMAS dataset, which contains different viewpoints. (c) "dial-phone" and (d) "eat-banana" actions from Rochester dataset, which contains complex daily activities.

The aim of our experiments is to examine CSoW model contribution, and the action recognition rates obtained for CSoW vs. the rates achieved by BoW. For this goal, in all experiments we kept the setup identical, and replaced only the model from BoW to CSoW. The suggested framework was tested on two public data-sets: IXMAS [21] and Rochester [14]. Figure 6 displays a few actions from these data-sets. For both data-sets, first each frame of each video sequence is represented using one of two tested descriptor-per-frame methods: (i) the view-invariant SSM descriptors of [8], and (ii) HOG descriptors [3]. Then, both CSoW and BoW representation are constructed. For IXMAS dataset we've also experimented with the descriptor-per-interest-point representation using STIP [9] descriptors.

We classify videos using a leave-one-actor-out configuration. Note that for the IXMAS dataset, all repetitions and all viewpoints of the same actor are extracted from the training set, since each actor performs each action 15 times is this dataset. The classification is done using SVM [1], with $kernel = e^{-\frac{distance}{\beta}}$, where $distance$ is the distance between CSoWs defined on section 4, and the value of the parameter β is set as

$$\beta = 2AVG(\text{training distance}) + STD(\text{training distance}). \quad (20)$$

We have experimented with many other kernels, all yielded similar results.

In all setups rigid CSoW was applied using a local neighborhood radius of $R = 7$, and loose CSoW was applied using a local neighborhood radius of $R = 25$, where the overlap between the preceding and the following histograms is of $lp = 2$. In section 6.3 we discuss the local neighborhood size effect. We used a codebook of 990 words for IXMAS dataset , and 300 for Rochester dataset.

Table 1 summarizes the action recognition rates for the different setups examined: 2 data-sets and 3 descriptor types. The results are presented for naive BoW, and the two CSoW options. It can be clearly seen that the CSoW achieves a significant improvement relative to BoW. Both BoW and CSoW implementations use the same descriptors and the same shared parameters such as code-book size etc. In some cases the Rigid-CSoW performed best, while in others Loose-CSoW did. It is important to note, that the differences between the results obtained by the two models are mild and in all cases both improve significantly over BoW.

6.1. SVM Kernel Parameter

In order to apply classification task, we use SVM, in libsvm [1] implementation. SVM requires a kernel matrix ker, which is based on the distances between samples, and is generally obtained by:

$$ker = e^{-\frac{distance^2}{\beta^2}} \quad or \quad ker = e^{-\frac{distance}{\beta}}, \quad (21)$$

where $\beta > 0$. We have examined the effect of β value on the recognition rates. The values of β examined were either functions of the training distance matrix statistics (average, median, STD of training distances), set values between $\beta = 10\% - 200\%$ of the maximal training distance, or adaptive β values based on the distance between the test sample and the training samples, as suggested in [22]. Fig. 7 displays the recognition rates for the different setups, for different $beta$ values and eq. 21. In addition to the fact that CSoW outperforms BoW significantly in most cases, CSoW is significantly more robust to β value, and preserves the results for different values of β, while BoW has many fluctuations in recognition rates vs. β value.

6.2. CSoW different components contribution

CSoW framework is composed of several stages. The first is adding the global context component (GCC), and the local context is added after labeling the descriptors using a codebook. A question which arises from these stages is what is the contribution of each stage to the total improvement achieved by using CSoW instead of BoW. This could be examined by isolating the global from the local context. Either by classifying with BoW, where the descriptors are the new descriptors, with GCC added (global context only), or alternatively - applying CSoW on descriptors without added GCC (local context only). Table 2 provides such an evaluation.

Table 1. Comparing CSoW and BoW.

Dataset	Descriptor	BoW	CSoW Rigid	CSoW Loose	Improvement Rigid CSoW	Improvement Loose CSoW
IXMAS	SSM	68.9	**71.9**	69.6	4.4%	1.1%
IXMAS	STIP	67.8	72.2	**72.7**	6.4%	7.1%
IXMAS	HOG	53.9	**69.9**	63.2	29.7%	17.3%
Rochester	SSM	30	56.7	**57.3**	88.9%	91.1%
Rochester	HOG	33.3	**52.7**	49.3	58%	48%

Table 2. Contribution of local vs. global context.

Dataset	Descriptor	Global only	Local only Rigid	Local only Loose	BoW	CSoW Rigid	CSoW Loose
IXMAS	SSM	**70.8**	69.8	69.6	68.9	71.9	69.6
IXMAS	STIP	68.3	71.4	**71.6**	67.8	72.2	72.7
IXMAS	HOG	**71.3**	59.3	59.3	53.9	69.9	63.2
Rochester	SSM	34.7	**51.3**	**51.3**	30	56.7	57.3
Rochester	HOG	40.7	**52.7**	50	33.3	52.7	49.3

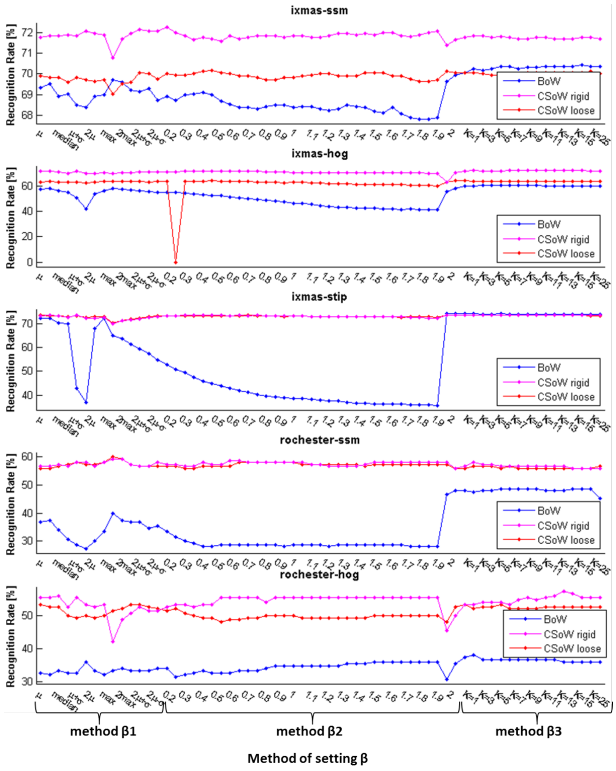


Figure 7. Recognition Rate vs. β for the different setups, eq. 21, non-squared distance.

It could be seen from the isolated results that both global and local context contribute to the overall improvement in recognition rates. In some cases the partition to two separate contributions is clear - the improvement obtained by either global context alone or local context alone is less than the one achieved by combining the whole framework. In other cases such as Rochester-HOG, it could be seen that the global context indeed improves the recognition rate, however the local context alone achieves the same improvement obtained by the combination of global and local context together, therefore in this case one can omit the global context usage. There are also cases probably in which global context is sufficient.

One of the possible reasons for the variety in the necessity of the different CSoW components could be the variety in human actions - some actions have periodic components, some tend to have a start-end pattern, some are composed of several sub-actions, some are fundamental actions, some have more variability in the sub-actions lengths inside the action, and so forth. Since the aim of the framework is to be as generic as possible, by examining the isolated global-local results and comparing to the overall results, the natural conclusion is to use the whole framework, unless we have some specific knowledge regarding the actions classified. In the generic case, the combined CSoW framework is most likely to achieve the best recognition rates.

6.3. Local Neighborhood's Size

In order to examine the robustness to the local neighborhood radius R, we applied CSoW with different R values. Figure 8 summarizes the results for different R values, for Rochester dataset with HOG descriptor. Similar behavior was observed for the other dataset-descriptor setups as well.

It can be seen that for both rigid and loose local context there is a wide range of radius values which provide similar recognition rates. Overly large or small radius breaks

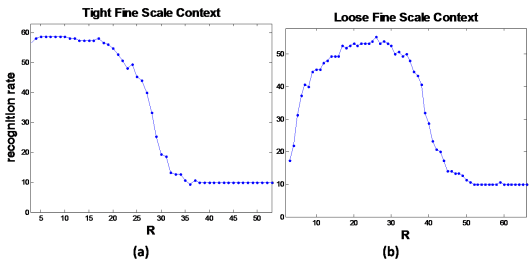


Figure 8. Recognition rates vs. R values, for Rochester datatset with HOG descriptor, for (a) Rigid CSoW and (b) Loose CSoW.

the locality assumptions. Loose-CSoW requires larger local neighborhood for two reasons: First, histogramming requires bigger amount of data to conclude statistical information. Second, in Rigid CSoW the local neighborhood refers to the number of parts surrounding a part, and in Loose CSoW the histograms are over codewords and not parts. We further tested variations of the overlap lp between values of 1-5, and found results were not changed.

7. Summary and Discussion

The Contextual Sequence of Words representation is a generalization of the Bag of Words model and for the case when temporal context has importance in the signal representation. In this work it was shown that CSoW can improve action recognition rates generally, for several setups, i.e., different data-sets and regardless of the features used. This is the main contribution of the CSoW model. We also note that CSoW could be directly applied, as is, to any signal whose temporal order and context is of significance, such as audio signals, text signals etc. It can also be extended to the spatial 2D domain, or even to the spatio-temporal 3D domain, in order to represent images and video signals.

8. Acknowledgements

This research was supported by Marie Curie IRG-208529, the Ollendorf foundation, and by the Fund for the Promotion of Research at the Technion.

References

[1] C.C. Chang, C.J. Lin, *LIBSVM : a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm 6

[2] G. Csurka, C. Dance, L. Fan, J. Willamowski, C. Bray *Visual categorization with bags of keypoints*, ECCV International workshop on statistical learning in computer vision,2004. 1

[3] N. Dalal, B. Triggs, *Histograms of oriented gradients for human detection*, In CVPR 2005, pp. 886-893. 1, 6

[4] F.J. Damerau. *A technique for computer detection and correction of spelling errors* Communications of the ACM, 1964. 5

[5] M. Fischler and R. Elschlager, *The representation and matching of pictorial structures*, IEEE Truns. on Computers, vol. 22, pp. 67-92,1973.

[6] R. Hamid, A. Johnson, S. Batta, A. Bobick, C. Isbell, G. Coleman. *Detection and Explanation of Anomalous Activities: Representing Activities as Bags of Event n-Grams*, In CVPR 2005. 1

[7] T. Hofmann, *Probabilistic latent semantic indexing*, 22nd international ACM SIGIR conference on Research and development in information retrieval, 1999.

[8] I. N. Junejo, E. Dexter, I. Laptev, P. Perez, *CView-independent action recognition from temporal self-similarities*, In PAMI,2010. 1, 6

[9] I. Laptev, T. Lindeberg, *Space-Time Interest Points*, In ICCV, 2003. 5, 6

[10] I. Laptev, P. Pérez, *Retrieving actions in movies*, Proc. ICCV, 2007. 1

[11] S. Lazebnik, C. Schmid, J. Ponce, *Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories*, In CVPR, 2006. 2

[12] J. Liu, M. Shah, *Learning human actions via information maximization*, In CVPR, 2008. 1

[13] F. Lv, R. Nevatia, *Single View Human Action Recognition using Key Pose Matching and Viterbi Path Searching*, In CVPR, 2007. 1

[14] R. Messing, C. Pal, H. Kautz, *Activity recognition using the velocity histories of tracked keypoints*, In ICCV, 2009 6

[15] J.C. Niebles, H. Wang, L. Fei-Fei, *Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words*, International Journal of Computer Vision (IJCV), 2008.

[16] M. Osadchy, E. Morash, *Loose shape model for discriminative learning of object categories*, In CVPR 2008 3

[17] S. Savarese, A. Del Pozo, JC. Niebles, L. Fei-Fei, *Spatial-Temporal Correlations for Unsupervised Action Classification*, IEEE Workshop on Motion and Video Computing, Copper Mountain, Colorado January 8-9, 2008. 1

[18] S. Savarese, J.Winn, A. Criminisi, *Discriminative Object Class Models of Appearance and Shape by Correlatons*, In CVPR, June 2006.

[19] M.M. Ullah, S.N. Parizi, I. Laptev *Improving Bag-of-Features Action Recognition with Non-local Cues*, in Proc. BMVC 2010, Aberystwyth, UK.

[20] H. Wang, M.M. Ullah, A. Klaser, I. Laptev, C. Schmid. *Evaluation of local spatio-temporal features for action recognition*, British Machine Vision Conference, 2009. 1

[21] D. Weinland, R. Ronfard, E. Boyer, *Free Viewpoint Action Recognition using Motion History Volumes*, Computer Vision and Image Understanding, 2006 6

[22] L. Zelnik-Manor, P. Perona *Self-Tuning Spectral Clustering*, Advances in Neural Information Processing Systems 17, pp. 1601-1608, 2005, (NIPS'04) 6