Human Activity Classification with Online Growing Neural Gas

Maximilian Panzner, Oliver Beyer, and Philipp Cimiano

Semantic Computing Group, CITEC, Bielefeld University,
obeyer@cit-ec.uni-bielefeld.de
http://www.sc.cit-ec.uni-bielefeld.de

Abstract. In this paper we present an online approach to human activity classification based on Online Growing Neural Gas (OGNG). In contrast to state-of-the-art approaches that perform training in an offline fashion, our approach is online in the sense that it circumvents the need to store any training examples, processing the data on the fly and in one pass. The approach is thus particularly suitable in life-long learning settings where never-ending streams of data arise. We propose an architecture that consists of two layers, allowing the storage of human actions in a more memory efficient structure. While the first layer (feature map) dynamically clusters Space-Time Interest Points (STIP) and serves as basis for the creation of histogram-based signatures of human actions, the second layer (class map) builds a classification model that relies on these human action signatures. We present experimental results on the KTH activity dataset showing that our approach has comparable performance to a Support Vector Machine (SVM) while performing online and avoiding to store examples explicitly.

Keywords: artificial neural networks, online growing neural gas, human action recognition, space-time, online classification

1 Introduction

The recognition and classification of human activity is important in many application domains including smart homes [1], surveillance systems [2], ambient intelligence [3], etc. In particular, we address the task of classifying human activity into a given set of activity types on the basis of video data, sequences of 2D images in particular. State-of-the-art approaches extract features from space-time volumes, e.g. *space-time interest points (STIP)* as introduced by Ivan Laptev [4]. Their advantage lies in their compact and robust representation of human actions, as they reduce the input space by identifying local feature points. Training a human action classifier model based on STIPs typically includes the clustering (with e.g. k-Means) of video features to yield bag-of-visual-words clusters that can be used to derive a histogram-based representation of a video clip by indicating the number of STIPs being assigned to each cluster. Then a classifier (e.g. SVM) is trained to learn to classify image sequences into a set of given human activity types.

There are several drawbacks in this classical approach. First of all, the approach requires an architecture that comprises heterogeneous algorithms, e.g. a feature extractor, a clustering algorithm and a classification algorithm. An architecture that is more uniform and compact relying on one algorithm would be simpler, easier to implement and thus preferable. Further, the approach is not online, requiring to store a number of examples in memory or on disk in order to recompute the cluster and retrain the classifier at regular intervals.

To circumvent these limitations, we present a new architecture which is based on Online Growing Neural Gas (OGNG), which has been presented earlier [5]. In this paper we present a two-layer architecture which consists of two maps that we call *feature map* and *class map*, respectively. In the first layer (feature map), STIPs are dynamically clustered according to their similarity in the feature space, yielding a growing set of "visual words" that can also change over time. A *human activity signature (HAS)* for each space-time volume is then formed by a histogram indicating the activity of each prototype / neuron in the feature map. In the second layer (class map), space-time volumes are clustered by their HAS and labelled according to the corresponding activity.

Our contributions can be listed as follows:

- **Online classification**: We provide an architecture that grows incrementally and is capable of processing space-time volumes in an online fashion as new data arrives.
- **Compact model**: We provide a compact human activity model as both layers are based on STIPs and OGNG which represent the high dimensional input space in a low dimensional map.
- **Uniform architecture**: We provide an architecture which is uniformly based on OGNG, in contrast to existing approaches that rely on more heterogeneous structures.

We compare our architecture to the classical architecture proposed by Laptev [4] based on a k-means based feature discretization as well as an SVM-based classification, showing that our approach yields comparable results to the latter approach, while proposing a uniform architecture based on two OGNG maps and circumventing the need to store examples to process them offline. Our approach is thus suitable in a life-long learning setting, in which there is a never-ending data stream that needs to be processed on-the-fly as in the applications mentioned above.

The paper is structured as follows: in Section 2 we describe the OGNG algorithm in order to make this paper self-contained. In Section 3 we describe our online approach to human action classification with OGNG, including a description of the features we use. In Section 4 our experiments, including the methodology of our evaluation, the used baseline and our results are described. We then conclude and provide an overview over the related work in Section 5.

2 Online Growing Neural Gas (OGNG)

Online Growing Neural Gas (OGNG) as introduced by Beyer and Cimiano [5], extends Growing Neural Gas to an online classifier by integrating additional online labeling and prediction strategies. In the following we will briefly describe the OGNG algorithm. The algorithm is depicted in Algorithm 1 and modifications are highlighted. A detailed description of OGNG and a comparison of several online labeling and prediction strategies can be found in Beyer and Cimiano. [5].

Algorithm 1 Online Growing Neural Gas (OGNG)

1: Start with two units i and j at random positions in the input space.
2: Present an input vector $x \in R^n$ from the input set or according to input distribution.
3: Find the nearest unit n_1 and the second nearest unit n_2.
4: Assign the label of x to n_1 according to the present labeling strategy.
5: Increment the age of all edges emanating from n_1.
6: Update the local error variable by adding the squared distance between w_{n_1} and x.

$$\Delta error(n_1) = |w_{n_1} - x|^2$$

7: Move n_1 and all its topological neighbours (i.e. all the nodes connected to n_1 by an edge) towards x by fractions of e_b and e_n of the distance:

$$\Delta w_{n_1} = e_b(x - w_{n_1})$$

$$\Delta w_n = e_n(x - w_n)$$

for all direct neighbours of n_1.
8: If n_1 and n_2 are connected by an edge, set the age of the edge to 0 (refresh). If there is no such edge, create one.
9: Remove edges with their age larger than a_{max}. If this results in nodes having no emanating edges, remove them as well.
10: If the number of input vectors presented or generated so far is an integer or multiple of a parameter λ, insert a new node n_r as follows:
Determine unit n_q with the largest error.
Among the neighbours of n_q, find node n_f with the largest error.
Insert a new node n_r halfway between n_q and n_f as follows:

$$w_r = \frac{w_q + w_f}{2}$$

Create edges between n_r and n_q, and n_r and n_f. Remove the edge between n_q and n_f.
Decrease the error variable of n_q and n_f by multiplying them with a constant α. Set the error n_r with the new error variable of n_q.
11: Decrease all error variables of all nodes i by a factor β.
12: If the stopping criterion is not met, go back to step (2).

In steps 1-3, the network is initialized and the first winner n_1 and second winner n_2, according to a presented stimulus, are determined. In step 4 we assign the label of our stimulus ξ to the winner neuron n_1 according to the selected labeling strategy. The selected labeling strategy is the *relabeling method (relabel)*, because of its simplicity and effectiveness as shown in Beyer and Cimiano [5]. In steps 5-7, the age of all edges emanating from n_1 are incremented by one. Furthermore, the local error gets updated, and n_1 and its topological neighbors n are adapted towards the stimulus by the learning rates e_b (for n_1) and e_n (for the topological neighbors). In steps 8-9, n_1 and n_2 get connected by an edge and

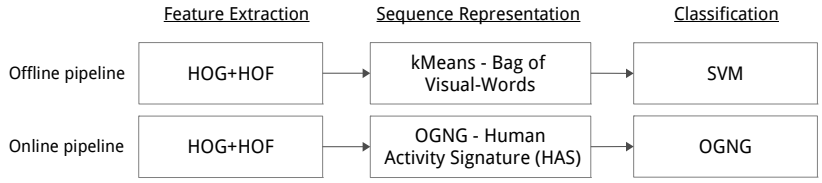

Fig. 1. Online and offline processing pipelines.

the edges with an age larger than a_{max} are removed. In step 10, a new neuron n_r is introduced between the neuron n_q with the largest local error and its neighbor n_f, having the largest error of its neighborhood. We only insert a new neuron if n_{max}, the maximal number of neurons per class, is not exceeded for the class of ξ. In step 11 all error variables are decreased by the factor β. In the last step 12, the algorithm continues with step 2 if the stopping criterion (mostly when a predefined maximum number of neurons has been reached) is met.[1]

3 Human Activity Classification with OGNG

In this section we describe our two-layer online approach to human activity classification that exploits topological maps - Growing Neural Gas maps in particular - at both layers. We call our approach *Online Human Action Classifier* (OHAC). A typical processing pipeline in human activity recognition is comprised of the following three steps:

1. **Extraction of video features:** Extraction of distinctive features from a sequence of video frames.
2. **Representation of video features:** Calculating video signatures from the extracted features that capture the similarity between different video sequences.
3. **Classification of video signatures:** Training a classifier to learn to recognize a set of previously observed classes of human actions.

Our algorithm covers steps 2 and 3 of the typical processing pipeline. For the first step, the extraction of video features, we use HOG+HOF features calculated around sparsely detected spatio-temporal interest points (STIPS) as proposed by Laptev [4]. The detection of the points and the extraction of the feature descriptors are completely online in the sense that no global information is required. STIPs are detected as local maxima of a Harris-Corner-Function extended into the spatio-temporal domain. The spacial Harris-Corner-Function characterizes the "'cornerness"' of an image point by the strength of its intensity gradients in all directions. The spatio-temporal Harris-Corner-Function responds to points in space-time where the motion of local image structures is non-constant. Aside from sensor noise and other disruptions, the motion of local image structures is

[1] For our experiments we additionally introduce new neurons for novel categories in the learning process. Furthermore, we only stop inserting new neurons when the maximum number of neurons is reached, instead of stopping the training process.

primarily the result of forces acting on the corresponding physical objects. Hence the local neighbourhood of the detected points can be expected to provide meaningful information about motion primitives in that point of space-time. The combination of STIPs and HOG+HOF feature descriptors has already shown promising results in several synthetic [4] and real world datasets [4, 6].

3.1 Static Human Action Classifier according to Laptev

As baseline we use an offline approach proposed by Laptev [4]. This approach uses k-means for clustering and an SVM for classification. The video sequence is represented as a *bag of visual words* histogram. The visual vocabulary is built by clustering the feature vectors in the training set into a predetermined number of clusters.

Algorithm 2 Static Human Action Classifier according to Laptev

1: Cluster the training set into a predetermined number of clusters using kMeans.
2: Initialize Histogram H with one entry for each prototype vector.
3: Present an input vector $x \in S$ from the extracted features of the video sequence.
4: Find the nearest prototype p_x to the presented input.
5: Increment the corresponding histogram entry p_x by one.
6: repeat step 3 until all features are processed.
7: Normalize H using L_1 norm
8: Train the SVM with the normalized histogram and the label l of the current sequence.

In step 1 the *visual vocabulary* is built in advance by clustering all feature vectors in the training set into a predetermined number of clusters[2] using kMeans. Each cluster stands with its prototype vector for one distinct *visual word* in the *visual vocabulary*. Steps 3-6 iterate over the feature vectors in the current training sequence. Each feature vector is assigned to its nearest prototype vector and incorporated into the histogram. The resulting histogram represents the given video sequence as a bag of visual words. To compensate for different counts of features in different sequences, the histogram is normalized using the L_1 norm. In step 8 the SVM is trained with the normalized histogram and the corresponding label of the current sequence.

3.2 Online Human Action Classifier (OHAC)

The Online Human Action Classifier consists of two independent OGNG networks. The first network is what we call the *feature map*. It utilizes the OGNG algorithm for clustering incoming data in feature space. The second network is called the *class map*, as it uses the label information from a training sequence to assign class labels to the nodes in the network according to the *relabel* strategy presented in section 2. Video sequences are represented as *human activity signatures* (*HAS*), which are represented by a histogram indicating the activity of each neuron in the *feature map*, while iterating through the respective video sequence.

[2] We use $k = 4000$ for the number of clusters following Laptev [6]

Algorithm 3 Online Human Action Classifier (OHAC)

1: Initialize the *feature-* and *class-maps*.
2: Initialize the HAS histogram H with two (number of initial nodes in the *feature map*) entries $h_1 = 0, h_2 = 0$.
3: Present an input vector $x \in S$ from the extracted features of the video sequence.
4: Find the nearest unit n_x from the *feature map*.
5: **if** node n_x is new **then**
6: insert new entry into the histogram at position x
7: **end if**
8: Increment the corresponding histogram entry h_x by one.
9: repeat step 3 until all features are processed.
10: Normalize H using L_2 norm.
11: Update the OGNG *class map* with the normalized HAS histogram H and label l of the video sequence.

Algorithm 3 is initialized by first initializing the OGNG network and creating an empty *bag of visual words* histogram (steps 1-2). The histogram starts with two entries, one for each of the two initial nodes in the OGNG network. In steps 3-5 the next input vector x is presented to the *feature map*, and the node n_x that is closest to the presented stimulus is located. If the located node n_x is newly inserted into the *feature map*, a new histogram entry is created at position x. In step 8 the histogram entry at position x is incremented. When all input vectors in the sequence are processed, the histogram is normalized by L_2 norm. The normalization compensates for different numbers of extracted feature vectors in different video sequences. The normalized histogram is then used to update the *class map* OGNG network with the label l of the video sequence. To predict the label of a previously unseen video sequence S, all input vectors $x \in S$ are incorporated into the histogram H by the number of the *feature map* node n_x nearest to them (steps 3-5). The histogram is then normalized and the label l is predicted by the *class map* according to the *single linkage strategy* (see Beyer and Cimiano [5]).

4 Experiments and Evaluation

4.1 Dataset

As a dataset for evaluation we use the KTH human action dataset [7] [3]. This video database consists of six categories of human actions (walking, jogging, running, boxing, hand waving and hand clapping). All actions are performed several times by 25 subjects in four different scenarios (outdoors, outdoors with scale variations, outdoors with different clothes and indoors). Each combination of 25 subjects, 6 actions in 4 scenarios gives a total of 600 video files.

4.2 Evaluation Methodology

We evaluated the accuracy of OHAC and our baseline on the KTH human action dataset. We generated 15 training and test sets by separating the 600 video clips

[3] Examples of the six actions are shown on the following website http://www.nada.kth.se/cvap/actions/

into 300 training examples and 300 test examples for each set. We furthermore took care that each of the six categories was equally distributed in the training and test set. We averaged our accuracy results over the 15 runs and also determined a best and worst result of the 15 runs.

The OGNG parameters are set as follows: insertion parameter $\lambda = 50$; maximum age $a_{max} = 120$; adaptation parameter for winner $e_b = 0.3$; adaptation parameter for neighbourhood $e_n = 0.0018$; error variable decrease $\alpha = 0.5$; error variable decrease $\beta = 0.0005$. We also allowed a maximum of 4000 neurons for the feature map and 200 for the class map.

4.3 Results

Our results are depicted in Table 1. The matrices show the confusion matrix of our Baseline (left) and OHAC (right). Thereby, each row represents the to be classified human action category, while each column holds the percentage of examples that have been classified into the category written on top of the matrices. Overall, OHAC achieves an averaged accuracy of 93%, while our Baseline holds an accuracy of 95%. We performed a t-test and could not prove that the results of both approaches are statistically significant. We thus consider the classification performance of the algorithms to be comparable. The confusion matrix shows that both algorithms are having issues to distinguish between jogging and running, which is intuitively understandable as we as humans also would consider those two activities to be closer to each other compared to the other four. Furthermore, it is interesting that OHAC slightly less confuses the human action categories of "hand clapping" and "running" with 93.2% and 72.2% compared to 89.8% and 66.8% of our Baseline. It also should be mentioned that the confusion of OHAC is spread more uniformly compared to the Baseline approach, which could be explained by the generative approach of OHAC compared to the discriminative character of our Baseline and i.e. of the underlying SVM classifier.

5 Related Work & Conclusion

In this paper we have presented a novel human activity classifier model based on Online Growing Neural Gas (OGNG). The model provides a compact architecture and consists of two layers, allowing the storage of human actions in a more memory efficient structure. While the first layer (feature map) dynamically clusters STIPs and serves as base for the creation of histogram-based signatures of a human action, the second layer (class map) builds a classification model that builds upon those human action signatures. The advantage of this novel architecture lies in its ability to perform a human action classification task online as the model stepwise adapts to new data and grows incrementally. The uniform character of the algorithm is desirable, as that its simplicity allows an easy implementation and integration into existing systems. In most cases, heterogeneous offline human action recognition approaches have been proposed [4,

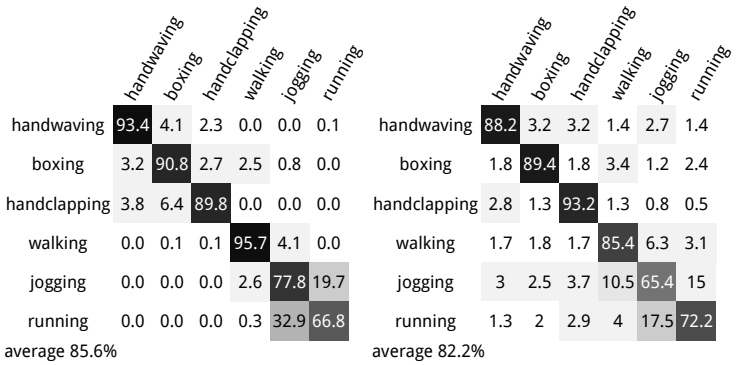

Table 1. Confusion matrix of our Baseline (left) and OHAC (right) on the KTH human action database, averaged over 15 runs.

8], that generate action signatures by clustering STIPs with static clustering algorithms (such as k-Means) and classifying them with an offline classifier that needs to be retrained as new data arrives. We have experimentally shown on the KTH dataset that our approach reaches comparable performance to a classical offline SVM-based classification approach while performing online and avoiding the need to store training examples explicitly, thus being suitable in lifelong stream data settings.

References

1. Marie Chan, Eric Campo, Daniel Estève, and Jean-Yves Fourniols. Smart homes-current features and future perspectives. *Maturitas*, 64(2):90–97, 2009.
2. M Valera and SA Velastin. Intelligent distributed surveillance systems: a review. In *Vision, Image and Signal Processing, IEE Proceedings-*, volume 152, pages 192–204. IET, 2005.
3. Eric J Pauwels, Albert Ali Salah, and Romain Tavenard. Sensor networks for ambient intelligence. In *Multimedia Signal Processing, 2007. MMSP 2007. IEEE 9th Workshop on*, pages 13–16. IEEE, 2007.
4. Ivan Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2-3):107–123, 2005.
5. Oliver Beyer and Philipp Cimiano. Online labelling strategies for growing neural gas. In *Intelligent Data Engineering and Automated Learning-IDEAL 2011*, pages 76–83. Springer, 2011.
6. Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008.
7. Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: a local svm approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 32–36. IEEE, 2004.
8. Alper Yilmaz and Mubarak Shah. Actions sketch: A novel action representation. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 984–989. IEEE, 2005.