A Deep Neural Architecture for Kitchen Activity Recognition

Roger Granada[*] and **Juarez Monteiro**[*] and **Rodrigo C. Barros**[†] and **Felipe Meneguzzi**[†]
School of Informatics - Pontifical Catholic University of Rio Grande do Sul
Av. Ipiranga, 6681, 90619-900, Porto Alegre, RS, Brazil
[*] {roger.granada,juarez.santos}@acad.pucrs.br
[†] {rodrigo.barros, felipe.meneguzzi}@pucrs.br

Abstract

Computer-based human activity recognition of daily living has recently attracted much interest due to its applicability to ambient assisted living. Such applications require the automatic recognition of high-level activities composed of multiple actions performed by human beings in a given environment. We propose a deep neural architecture for kitchen activity recognition, which uses an ensemble of machine learning models and hand-crafted features to extract more information of the data. Experiments show that our approach achieves the state-of-the-art for identifying cooking actions in a well-known kitchen dataset.

1 Introduction

Effective assistive applications require accurate identification of the activities being performed by the user being helped. Here, activity recognition refers to the task of dealing with noisy low-level data directly from sensors (Sukthankar et al. 2014). Failure to correctly identify the activity a user is performing has a cascade effect that often leads to users being frustrated and giving up on an assistive application. Such task is particularly challenging in the real (physical) world, since it either involves fusing information from a number of sensors or inferring enough information using a single sensor.

Single-sensor activity recognition often relies on a video camera feed (Karpathy et al. 2014), which until recently has posed a challenging research goal in computer vision. Advances in hardware and greater availability of data have allowed deep learning algorithms in general, and Convolutional Neural Networks (CNNs) in particular, to consistently improve on the state-of-the-art. CNNs achieve the state-of-the-art results when dealing with image-based tasks such as object recognition, detection, and semantic segmentation (Krizhevsky, Sutskever, and Hinton 2012; Long, Shelhamer, and Darrell 2015). Encouraged by those results, novel approaches use deep neural architectures to perform video-based tasks (Karpathy et al. 2014).

In this paper, we address the problem of recognizing human activities in an indoor environment with a single static camera. Our contribution focus on supporting people when they are in the kitchen, with the final goal of recognizing their actions when cooking meals. Our approach relies on a deep neural architecture that comprises multiple convolutional neural networks that are fused prior to performing the action classification. We perform experiments using the Kitchen Scene Context based Gesture Recognition dataset (KSCGR) (Shimada et al. 2013), and we show that our proposed approach outperforms the current state-of-the-art method (Bansal et al. 2013) for this particular dataset.

This paper is organized as follows. Section 2 provides a background on artificial neural networks and deep learning. Section 3 details our novel deep neural architecture for action recognition, whereas Section 4 presents a thorough experimental analysis for assessing the performance of our proposed approach. Section 5 points to related work and we finish this paper with our conclusions and future work directions in Section 6.

2 Background

Ordinarily, machine learning algorithms such as artificial neural networks (ANN) have been used to support many challenges of activity recognition. Conventional machine-learning techniques were limited in their ability to process natural data in their raw form (LeCun, Bengio, and Hinton 2015). For decades, constructing machine learning systems required considerable domain expertise to create an internal representation (feature construction (Sondhi 2009)) from which the learning subsystem could detect or classify patterns in the input. Deep learning such as convolutional neural networks mitigates this problem by automatically learning representations, such that representations are expressed in terms of hierarchical features, allowing the computer to build complex concepts out of simpler concepts. This hierarchical representation allows a machine to be fed with raw data and to automatically discover the representations needed for detection or classification tasks. CNNs use these feature hierarchies for learning different representations from unstructured data such as images, videos and audio (Krizhevsky, Sutskever, and Hinton 2012). They have been shown to accurately classify images (Simonyan and Zisserman 2014b) and videos (Karpathy et al. 2014). However, CNNs have some limitations such as overfitting (due to the large number of parameters), which makes it challenging to to create a knowledge model that is capable of generalizing and accurately classifying unseen data.

3 Architecture Design

In this paper, we develop a deep neural architecture for action recognition in indoor environments with a fixed camera. Our architecture has three main components: i) data pre-processing, ii) convolutional architecture for action recognition; and iii) fusion strategies for the final classification.

Figure 1 illustrates the pipeline of our architecture: *RGB* represents the regular dataset with the RGB video frames; *OFL* represents the dataset generated by dense optical flow; *CNN* represents the Convolutional Neural Network architecture we use to recognize activities; *NN* is a neural network that weights the contribution of the probabilities generated by the output of two previous CNNs; *Mean* computes the arithmetic mean of the classification probability output from the two previous CNNs; and *SVM* is a Support Vector Machine classifier with linear kernel that classifies the output vectors from the two previous CNNs.

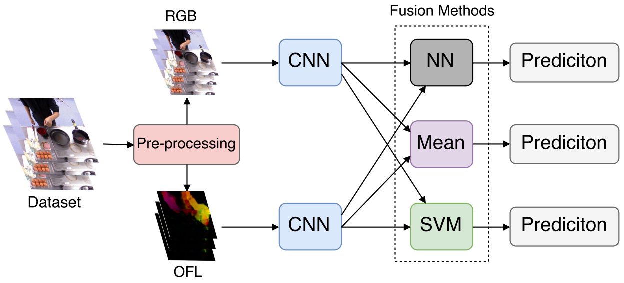

Figure 1: Pipeline of our methodology for activity recognition.

The pipeline starts receiving images from our dataset and pre-processing them, generating optical flow and RGB images with resolution of 256×256. Each resized image feeds a convolutional neural networks (CNN) that yields the values of probability to each class. Each fusion method (*NN*, *Mean* and *SVM*) receives these values of probability from CNNs and predicts the class to the input image. In what follows, we further detail each of the components of the proposed architecture.

3.1 Data pre-processing

Pre-processing consists of two steps: image resizing and optical flow generation. Resizing is important because it reduces the number of features for each image inside the CNN as well as the processing time. This step resizes all images of the dataset to a fixed resolution of 256×256. The second step generates the dense optical flow (Farnebäck 2003) of each pair of images. Optical flow represents the 2D displacement of pixels between frames generating vectors corresponding to the movement of points from the first frame to the second. A dense optical flow generates these displacement vectors, *i.e.*, vectors for horizontal and vertical displacements, for all points within frames. In order to generate the final image for each sequence of frames, we combine the 2-channel optical flow vectors and associate color to their magnitude and direction. Magnitudes are represented by colors and directions by hue values. Two new datasets compose the output of the pre-processing component: the original data with RGB channels and resized size (hereafter called RGB dataset), and the optical flow data representing the motion across frames (hereafter called OFL dataset).

3.2 CNN Architecture

Although a number of off-the-shelf CNN architectures are available (Simonyan and Zisserman 2014a; 2014b), in this work we develop an architecture based on inception modules (Szegedy et al. 2015), due to their top performance and, at the same time, reduced number of parameters. The proposed architecture is 22-layer deep and its inception modules contain convolutional filters in different scales/resolutions, covering clusters of diverse information. The network receives video frames as input that goes through several convolutional layers, pooling layers and fully-connected layers (FC). After a *Softmax* layer, the network outputs a vector containing the probability of the image for each class. In our pipeline, we use the same CNN architecture and training parameters to process both RGB and OFL datasets.

3.3 Fusion Methods

As the output of each CNN yields a vector containing the probability scores for each class, our model architecture allows for the application of three different fusion methods. The fusion methods intend to merge these vectors in order to increase the accuracy for the action recognition task. Figure 1 shows the 3 different approaches: i) a neural network (*NN*) that weights the contribution of the two probability vectors, ii) the standard arithmetic mean, *i.e.*, weight 0.5 for both vectors (*Mean*), and iii) a multi-class linear Support Vector Machine (*SVM*) (Crammer and Singer 2001).

The *NN* fusion contains a one-layer neural network to optimize the weights of the probabilities derived from the output of both CNNs. Figure 2 illustrates the structure of this neural network, where w_1 and w_2 are learned weights, $[A]$ is the vector containing the probabilities from the output of the CNN that processes the optical flow images, $[B]$ is the vector containing the probabilities for each class generated by the output of the CNN that processes the RGB images, and $[C]$ is the vector containing the weighted mean for each class. The idea behind this neural network is that its weights (w_1 and w_2) can be learned automatically by minimizing a loss function and backpropagating the gradients. During test time, this fusion method uses the learned weights to identify the class that has the greatest weighted mean. The *Mean* fusion receives the output vector from both RBG and OFL CNNs and calculates the arithmetic mean for each class, assigning to the image the class with the highest score. The *SVM* fusion contains a multi-class linear Support Vector Machine trained with the CNN output from the validation dataset. At test time, the SVM predicts the class with the largest score.

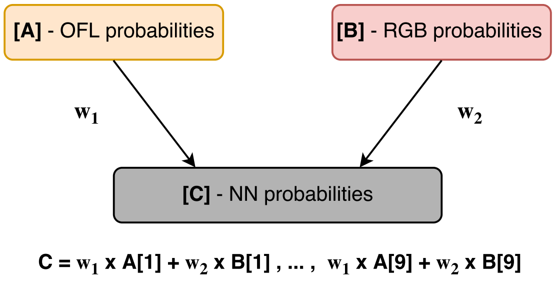

Figure 2: Single-layer neural network developed to compute the weighted mean from the outputs of two CNNs.

4 Experimental Analysis

In this section, we describe the dataset used in our experiments for indoor fixed-camera activity recognition and the implementation details used in the CNN models and fusion methods.

4.1 KSCGR Dataset

The Kitchen Scene Context based Gesture Recognition dataset[1] (KSCGR)(Shimada et al. 2013) is a fine-grained kitchen action dataset released as a challenge in ICPR 2012[2]. The dataset contains five menus for cooking eggs in Japan: *ham and eggs*, *omelet*, *scrambled egg*, *boiled egg*, and *kinshi-tamago*. A total of 7 different subjects perform each menu. The ground truth data contains the frame id and the action being performed within the frame. There are 8 cooking gestures in the dataset: *breaking*, *mixing*, *baking*, *turning*, *cutting*, *boiling*, *seasoning*, *peeling*, and *none*, where *none* means that there is no action being performed in the current frame or that the current action does not fit in any other classification. Figure 3 illustrates example frames for each class and the distribution of all classes in the dataset.

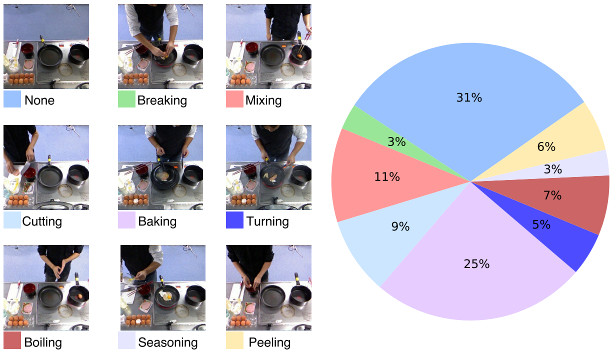

Figure 3: Activities of the KSCGR dataset and their percentage of the total number of frames in the dataset.

We divided the dataset into training, validation and test sets. The training set contains 4 subjects, each of them performing 5 recipes, *i.e.*, 20 videos and 139,196 frames in total. We use the validation set to obtain the model configuration that best fits the training data, *i.e.*, the configuration with the highest accuracy. This set contains 1 subject performing 5 recipes with 32,897 frames in total. We use the test set to assess the accuracy of the selected model in unseen data. It contains 2 subjects, each performing 5 recipes, *i.e.*, 10 videos with 55,781 frames in total.

4.2 Implementation

CNN architecture: In order to perform the activity recognition task we use an inception-based CNN architecture (Szegedy et al. 2015). The training phase uses mini-batch stochastic gradient with momentum (0.9). For each iteration, the network forwards a mini-batch of 128 samples. Before passing through the the layers of the network the CNN applies a random crop, *i.e.*, a crop in a random part of the image, and a random horizontal flip in the input image, generating a sub-image of 224×224. All images have their pixels subtracted by the mean pixel of all training images. All convolutions, including those within the inception modules, use rectified linear activation functions. Regarding weight initialization, we employ the *Xavier* algorithm that automatically determines the value of initialization based on the number of input neurons. To reduce the chances of overfitting, we apply dropout on the fully-connected layers with a probability of 70%. The learning rate is set to 10^{-3} and we drop it by a factor of 50 every epoch, stopping the training after 43.5k iterations (30 epochs).

NN: We train the neural network of the fusion method with data from the validation set for 10 epochs with weights w_1 and w_2 initialized with 0.5. We use a mean squared error loss function and optimize it through the Adam (Kingma and Ba 2014) adaptive learning rate method with the same learning rate used to train the CNN.

SVM: We train the multi-class Support Vector Machine using the off-the-shelf implementation by Crammer and Singer (2001) from *scikit-learn*[3] toolbox. As the neural network fusion, we train the SVM using the validation set. We use the linear kernel and default *scikit-learn* regularization parameter $C = 1$ with the square of the *hinge loss* as loss function.

4.3 Results

In order to evaluate our approach, we compare the output of each fusion method in the test set. We use the classification generated by each individual CNN as baseline, and thus, we can see whether the fusion method improves over each individual CNN. Table 1 shows the accuracy scores for each class individually (*None*, *Breaking*, *Mixing*, *Baking*, *Turning*, *Cutting*, *Boiling*, *Seasoning*, *Peeling*) and the global accuracy (*All*) that considers all classes at once. Rows *RGB* and *OFL* contain the accuracy for the baselines and *Mean*, *SVM* and *NN* rows contain the accuracy for the fusion methods.

As we can observe in Table 1, the fusion neural network (*NN*) achieves the best results in 5 out of 9 activities (*None*,

[1] http://www.murase.m.is.nagoya-u.ac.jp/KSCGR/

[2] http://www.icpr2012.org/

[3] http://scikit-learn.org/

Mixing, *Baking*, *Boiling* and *Seasoning*), obtaining a global accuracy (*All*) of 72.6%. Besides for *Turning*, the fusion methods improved the results for all other activities indicating that using a fusion method tend to improve results. A possible reason for the low performance of the fusion methods for classifying *Turning* might be a mixing of the limited number of frames for this activity and the training phase using vector probabilities generated with the validation set. Since our fusion methods are trained with predicted probabilities from validation set, any misclassification may lead to errors in training.

Since classification accuracy takes into account only the proportion of correct results that a classifier achieves, it is not suitable for unbalanced datasets as it may be biased towards classes with larger number of examples. By analyzing the KSCGR dataset, we note that it is indeed unbalanced, *i.e.*, classes are not equally distributed over frames. Figure 3 shows the distribution of frames per classes in the dataset. For instance, the *None* class has the largest number of frames ($\approx 30\%$ of the total) followed by *Baking* ($\approx 25\%$ of the total), while *Breaking* contains only $\approx 3\%$ of the frames. For dealing with the unbalanced nature of the KSCGR dataset, we measure the performance of the fusion methods based on Precision (P), Recall (R) and F-Measure (F).

Table 2 shows the values of Precision, Recall, F-measure and Accuracy achieved by the baselines (*RGB* and *OFL*) and the fusion methods (*Mean*, *SVM* and *NN*). In order to compare with the current state-of-the-art in the KSCGR dataset, in Table 2 we present the results of the hand-crafted features (*HCF*) proposed by Bansal *et al.* (Bansal et al. 2013), and also their results after a post-processing step (*HCF+PP*).

Note that *NN* fusion methods achieve the best scores for all measures. A large precision value means that the respective model can adjust very well to the features for identifying the class, whereas low values indicate that it cannot extract relevant features to identify the correct class among the remaining classes. The individual RGB CNN achieves 27% of accuracy for the *Breaking* class and 67% of accuracy for the *Baking* class, meaning that the features of the class *Breaking* are not so evident as the features of *Baking*. A possible reason for that the small number of training examples given to the models from the *Breaking* class. *Baking*, on the other hand, is much more present within the dataset, improving the training experience and making the neural architecture generalize better for frames that belong to that class. Even without a fusion method, individual RGB CNN shows the improvement when using CNNs for activity recognition when compared with hand-crafted features used by Bansal *et al.* (Bansal et al. 2013).

The normalized confusion matrix depicted in Figure 4 shows the effect of the *NN* fusion method, where rows represent the predicted classes and columns the true classes. Shades of blue represent the value in each cell, going chromatically from a darker blue for higher values to a lighter blue for lower values. The confusion matrix shows normalized values, *i.e.*, predicted values are divided by the total number of true values for each cell.

By analyzing the results for the *Breaking* class in Figure 4, we can see that the system incorrectly predicts it as *Peeling*.

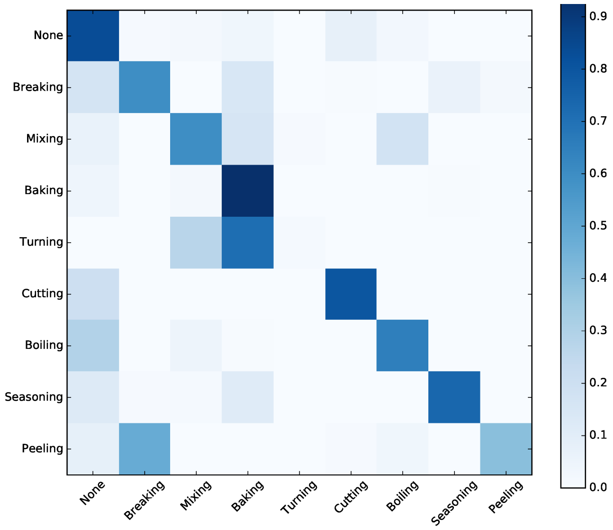

Figure 4: Normalized confusion matrix for the neural network (*NN*) fusion method.

Such misclassification makes sense since both activities occur in the same region of the frame using the same objects (*e.g.*, in both activities the subject is working on the left side of the scene and whereas in *Breaking* the subject breaks the egg onto the bowl letting the white egg and yolk fall down, in *Peeling* the subject peels the egg on the bowl letting the eggshell fall into the bowl.

Even though both *None* and *Baking* classes have higher values in the main diagonal of the confusion matrix, their precision scores are reduced by the misclassification of other classes. *Baking*, for instance, is classified many times as *Turning*. This misclassification shows that the CNN could not learn features that differ both classes. As the misclassification of *Breaking* and *Peeling*, the *Baking* and *Turning* activities occur in the same region of the scene and with the same objects.

Unlike *Baking* that does not have many changes through frames (*e.g.*, the egg baking inside the pan), the *None* activity is labeled as anything that happens but the 8 activities, encompassing frames in which the subject is preparing the kitchen utensils, moving pans, and inter-activity frames such as removing the egg from boiling to peeling. The large accuracy (69%) for classifying *Baking* may be due to this standard behavior with low variability in regions of the scene and the larger number of frames. Despite the unbalanced nature of the dataset, the values of accuracy follow the behavior of the F-Measure scores, where the lowest value is obtained for *Turning* and the largest value for *Baking*.

Since the process of identifying activities occurs frame by frame instead of a sequence of frames, sometimes the misclassification of a small number of frames in an activity may occur. For example, the misclassification of 8 frames of the *Baking* activity in the middle of $\approx$ 200 frames of the *None* activity. Following the work of Bansal *et al.* (2013), we apply a smoothing process on the output sequence of frames in order to remove the frames that are probably incorrectly

Table 1: Per-activity accuracy in the KSCGR dataset for all baselines and fusion methods.

Method	None	Breaking	Mixing	Baking	Turning	Cutting	Boiling	Seasoning	Peeling	All
RGB	0.644	0.275	0.289	0.671	**0.346**	0.588	0.287	0.363	0.117	0.689
OFL	0.519	0.341	0.314	0.600	0.194	0.545	0.128	0.382	0.449	0.631
Mean	0.634	0.327	0.340	0.684	0.174	**0.620**	0.169	0.403	0.347	0.692
SVM	0.679	**0.357**	0.432	0.689	0.000	0.526	0.444	0.601	**0.455**	0.721
NN	**0.690**	0.354	**0.452**	**0.693**	0.012	0.516	**0.505**	**0.651**	0.382	**0.726**

Table 2: Precision, recall, F-Measure, and accuracy for all baselines, fusion methods and the current state-of-the-art approach for the KSCGR dataset.

Approach	Precision	Recall	F-measure	Accuracy
HCF (Bansal et al. 2013)	0.62	0.63	0.61	0.64
HCF+PP (Bansal et al. 2013)	0.68	0.68	0.68	0.72
RGB (ours)	0.69	0.68	0.69	0.69
OFL (ours)	0.64	0.63	0.63	0.63
Mean (ours)	0.71	0.69	0.70	0.69
SVM (ours)	0.67	0.72	0.70	0.72
NN (ours)	**0.72**	**0.73**	**0.72**	**0.73**

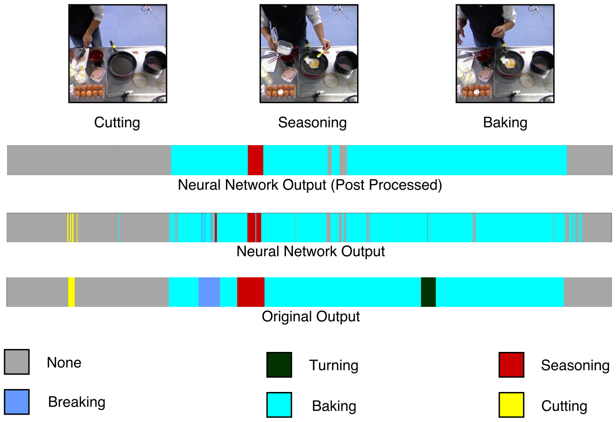

Figure 5: Temporal representation of classes for the frame sequence of a single video in which true labels are compared with labels predicted by our approach.

classified. For instance, a frame in the middle of a sequence of 30 frames that contains a different class probably suggests that the frame was misclassified, given that activities would not occur in a single frame. To perform smoothing within the output classes, a window of fixed size slides through frames assigning to the target frame (the frame in the center of the window) the majority voting of all frames within the window.

Figure 5 presents a temporal representation of the distribution of classes in the frame sequence for a single video of the test set. Classes are represented by colored vertical lines in a temporal sequence for the original output (true labels), for the output of the neural network fusion method (*NN*), and for the *NN* output after smoothing. Note that in the original output the sequence of frames contains the subject preparing kitchen utensils (*None* class – gray lines), next a *Cutting* activity is performed (yellow lines), and the activities continue until the end of the current recipe.

By analyzing the output of *NN* in Figure 5, we can see that some frames are misclassified such as a single *Baking* action in the middle of the *None* class. After performing the post-processing smoothing step in the output, those noisy frames disappear. On the other hand, some frames that were correctly classified (*e.g.*, with the *Cutting* class), also disappear. Despite the increase in accuracy (from 86% to 88% for the example presented in Figure 5), the smoothed output completely ignored the existence of some activities. With that in mind, we preferred not to use any smoothing method in the pipeline of our neural architecture.

5 Related Work

Before the advent of CNN and neural networks in general, approaches used to recognize activities based on complex hand-crafted features extracted from video sequences (Bansal et al. 2013). Convolutional Neural networks on the other hand, learn automatically a hierarchy of features automating the process of feature construction. Thus, many authors propose to mix the still images with information encoded by hand-crafted features using CNNs (two-stream CNNs) (Simonyan and Zisserman 2014a). Recent research encodes the temporal dimension performing 3D convolutions in the convolutional layers of CNNs (3D CNNs) (Ji et al. 2013) capturing features along both the spatial and temporal dimensions. Although such approaches recognize activities, they are applied in other datasets and are not comparable to our work.

Bansal *et al.* (Bansal et al. 2013) perform daily life cooking activity recognition based on hand-crafted features for hand movements and object use in the KSCGR dataset (Shimada et al. 2013). Their method first detects hand regions through color segmentation and skin identification. They consider that objects may give hints of the activity, thus, identifying objects as "*Not in use*" and "*In use*". A hybrid model that combines a dynamic Support Vector Machine (SVM) and a Hidden Markov Model (HMM) considers both the structural and temporal information to jointly infer the activity, achieving 64% of accuracy. In order to improve the performance of the system, they perform a post-processing step of the output, removing noisy frames, *i.e.*, frames that are incorrectly classified among a cluster of correctly classified frames. Since some activities are temporally dependent of others (*e.g.*, *Peeling* only occurs after *Boiling*), they create a context grammar to select the the most likely guess for misclassified frames. Note, however, that making use of a human-made context grammar may not be an option in several real-world applications. Nevertheless, by using this post-processing step, Bansal *et al.* increase in $\approx 7\%$ the accuracy for the activity recognition, reaching 71%.

Ni, Moulin, and Yan (2015) propose an adaptive motion feature pooling scheme that utilizes human poses as side information. They extract hand-crafted features from the images, such as histogram of oriented gradient, motion boundary histogram, histogram of optical flow and trajectory shape in order to obtain more relevant features. The principal component analysis (PCA) algorithm reduces the dimension of the extracted features. Improved Fisher vectors encode the resulting features and a second application of PCA reduces once again the dimensionality. Finally, they train a Linear SVM in order to classify video segments. They perform experiments using two datasets, the KSCGR dataset (Shimada et al. 2013) and the MPII kitchen activity dataset (Rohrbach et al. 2012). Since their work only focus on object detection and tracking of movements, they do not present specific results for activity recognition, preventing us of performing a fair comparison with their work.

6 Conclusions and Future Work

In this work we developed a novel neural architecture for indoor fixed-camera kitchen activity recognition based on static and temporal data encoding using different fusion methods. The pipeline of the architecture includes the training of deep inception-based convolutional neural networks to extract features from images and classify unseen frames. Using optical flow and RGB frames from the kitchen scene dataset (KSCGR), we perform experiments showing that the convolutional networks can indeed learn high-level relevant features for the activity recognition task at hand. Experiments show that our approach that employs fusion methods achieve better results when compared with the current state-of-the-art work that employs only hand-crafted features (Bansal et al. 2013) or when compared with deep approaches that make use of RGB/optical flow images alone.

As future work, we intend to explore other hand-crafted features such as histogram of gradients (HOG), motion boundary history (MBH) and dense trajectories to extract more features from unbalanced data and feed them to deep convolutional neural networks. We also plan to employ other deep learning architectures such as Long-Short Term Memory networks (LSTM) (Hochreiter and Schmidhuber 1997) and 3D CNNs (Ji et al. 2013) considering that they are also capable of encoding temporal features, and use transfer learning in order to achieve better results.

Acknowledgement

This paper was achieved in cooperation with HP Brasil Indústria e Comércio de Equipamentos Eletrônicos LTDA. using incentives of Brazilian Informatics Law (Law nº 8.2.48 of 1991).

References

Bansal, S.; Khandelwal, S.; Gupta, S.; and Goyal, D. 2013. Kitchen activity recognition based on scene context. In *Proceedings of the ICIP 2013*, 3461–3465.

Crammer, K., and Singer, Y. 2001. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of machine learning research* 2(Dec):265–292.

Farnebäck, G. 2003. Two-frame motion estimation based on polynomial expansion. In *Scandinavian Conference on Image Analysis*, 363–370. Springer.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Ji, S.; Xu, W.; Yang, M.; and Yu, K. 2013. 3D convolutional neural networks for human action recognition. *IEEE TPAMI* 35(1):221–231.

Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; and Fei-Fei, L. 2014. Large-scale video classification with convolutional neural networks. In *Proceedings of the CVPR 2014*.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Proceedings of the NIPS 2012*, 1097–1105.

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *Nature* 521(7553):436–444.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the CVPR 2015*, 3431–3440.

Ni, B.; Moulin, P.; and Yan, S. 2015. Pose adaptive motion feature pooling for human action analysis. *International Journal of Computer Vision* 111(2):229–248.

Rohrbach, M.; Amin, S.; Andriluka, M.; and Schiele, B. 2012. A database for fine grained activity detection of cooking activities. In *Proceedings of the CVPR 2012*, 1194–1201.

Shimada, A.; Kondo, K.; Deguchi, D.; Morin, G.; and Stern, H. 2013. Kitchen scene context based gesture recognition: A contest in ICPR2012. In *Advances in depth image analysis and applications*. Springer. 168–185.

Simonyan, K., and Zisserman, A. 2014a. Two-stream convolutional networks for action recognition in videos. In *Proceedings of the NIPS 2014*, 568–576.

Simonyan, K., and Zisserman, A. 2014b. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sondhi, P. 2009. Feature construction methods: a survey. *sifaka.cs.uiuc.edu* 70–71.

Sukthankar, G.; ; Geib, C.; ; Bui, H. H.; ; Pynadath, D. V.; ; and Goldman, R. P. 2014. *Plan, Activity, and Intent Recognition*. Boston: Morgan Kaufmann.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proceedings of the CVPR 2015*.