Identification of Cooking Preparation Using Motion Capture Data: A Submission to the Cooking Activity Recognition Challenge

Clément Picard, Vito Janko, Nina Reščič, Martin Gjoreski, Mitja Luštrek

Abstract The Cooking Activity Recognition Challenge tasked the competitors with recognizing food preparation using motion capture and acceleration sensors. This paper summarizes our submission to this competition, describing how we re-ordered the training data, re-labeled it and how we hand-crafted features for this dataset. Our classification pipeline first detected basic user actions (micro-activities), using them it recognized the recipe, and then used the recipe to refine the original micro-activities predictions. After the post-processing step using a Hidden Markov Model, we achieved the competition score of 95% on the training data with cross-validation.

1 Introduction

Being able to perform basic daily activities such as cooking, dressing, bathing, and transferring in and out of a bed is essential to older adults' quality of life and health. Loss of independence in these activities is strongly associated with higher use of health services, nursing home placement, and death [1].

Ambient assisted technology can support older adults in performing their daily activities, for example by giving them advice, reminding them of key tasks or calling for help if needed. An important first step to do so, however, is to recognize these activities. This can be accomplished by wearable and ambient sensors combined with machine learning, a combination that is already adopted in some domains [2].

The Cooking Activity Recognition Challenge [3] aims to further push the state of the art, specifically in the domain of using wearable and ambient sensors for the recognition of the food preparation. The organizers created a dataset [4, 5]

Clément Picard
École normale supérieure de Rennes and Jožef Stefan Institute, e-mail: clement.picard@ens-rennes.fr

Vito Janko
Jožef Stefan Institute, e-mail: vito.janko@ijs.si

captured by wearable accelerometers and motion capture (mocap) sensors, and the competitors had to detect both the recipe being prepared and the micro-activities within this preparation (e.g. cutting, taking, washing, peeling etc.). One could then use such a system to detect if the observed user is regularly cooking or if they are doing some obvious mistakes in the preparation (e.g. skipping an important step).

Each of these sensors is suitable for the recognition of human activities [6, 7]. Having both of them available thus offers an interesting insight into how they compare, and should allow highly accurate recognition of food-preparation activities.

1.1 Method overview

The summary of our method is presented in Figure 1. First, we ordered the data segments that were originally shuffled. This allowed us to take into account temporal dependencies between different micro-activities. Then we used the mocap data to visualize the micro-activities in the training set and added hand-made labels to them. From the same data we then derived additional sensor streams (e.g. velocity of different body parts) and from them calculated a wide array of different features.

The classification process begun with one micro-activity classifier that could make first, rough, predictions. Using these predictions we could infer the underlying recipe of each sequence. This allowed us to use a specialized micro-activity classifier for each of the three different recipes. More precisely, we used two different classifiers for each recipe and then merged their predictions.

The final step was using a Hidden Markov Model (HMM) to smooth out the predictions. This model can learn the expected sequence of micro-activities for each recipe and can thus correct parts of sequences that look very atypical – most likely due to a misclassification error.

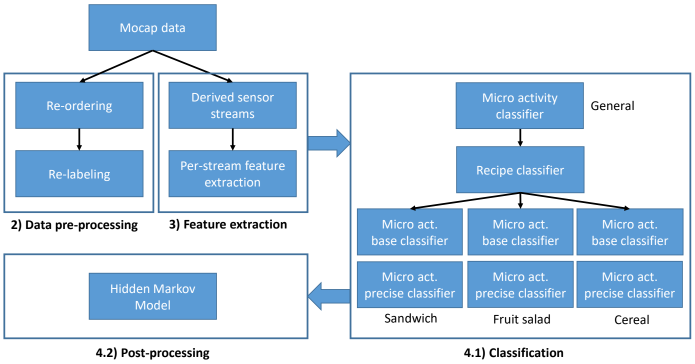
Mocap data
Re-ordering
Re-labeling
2) Data pre-processing
Derived sensor streams
Per-stream feature extraction
3) Feature extraction
Micro activity classifier
General
Recipe classifier
Micro act. base classifier
Micro act. base classifier
Micro act. base classifier
Micro act. precise classifier
Micro act. precise classifier
Micro act. precise classifier
Sandwich
Fruit salad
Cereal
4.1) Classification
Hidden Markov Model
4.2) Post-processing

Fig. 1 The pipeline for the proposed method. The step numbers corresponds to the numbers of the sections that describe them.

2 Challenge data

The Cooking Activity Recognition Challenge presented us with the data captured by four subjects, preparing three different recipes (*Sandwich*, *Fruit salad* and *Cereal*), five times each. Each recipe was composed of different actions (e.g. *Cut*, *Peel*, etc.) that we will call *micro-activities*. There were 10 different micro-activities in total. All subjects had to follow the same script for preparing each recipe, resulting in very similar sequences of micro-activities. The training dataset contained the data of three subjects and the appropriate labels, with the labeling of the last subject's data being the goal of the competition.

The presented dataset had data of two different types: mocap data collected using a set of 29 markers, and acceleration data from four worn devices – two smartphones positioned on the right arm and left hip, and two smartwatches on both wrists.

The mocap data had a sample rate of 100 Hz and no missing data. On the other hand, the sample rate of the acceleration data was different between the devices and also frequently varied during the recording. The average sampling rate was around 100 Hz for the smartwatches, and 50 Hz for the smartphones. In addition, the accelerometer data had several gaps, with 20–80% of the data missing, depending on the device.

After recording, the organizers segmented the data into 30-second segments. Each segment was then given two labels: the recipe being performed and the list of all micro-activities that happened in that segment. Notably, the start and end of each micro-activity was omitted. These segments were then shuffled and their original order was not given to the competitors.

2.1 Data pre-processing

The micro-activity aggregation (only having a list of micro-activities, not their time) presents a big problem for classical machine-learning methods that expect a single label for each time window. In addition, the shuffling complicates the recipe recognition, as it is hard to determine the recipe when seeing only a part of it. For example, taking an ingredients from the cupboard (*Take* micro-activity) is the same no matter what preparation procedure follows. To solve both problems, we pre-processed the dataset to make it look more "standard" and more similar to a dataset that we would acquire in a real-life setting.

First, we reordered all the segments to their original order. To do so, we leveraged the fact that if two segments are subsequent in a recording, then the end of one segment must be very similar to the beginning of the next one. We calculated the difference between the values of mocap markers between the beginning and the end for each pair. This was done for the x, y, z coordinates for all 29 marker values, and then all the differences were summed together. The pairs with the smallest differences indicated subsequent segments. If a segment did not have any that preceded it, it

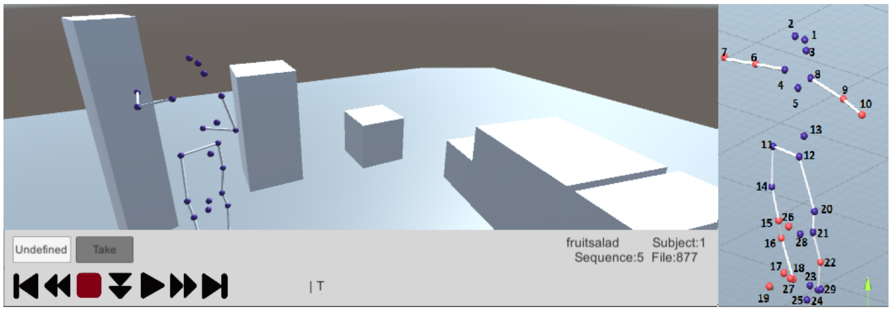

Fig. 2 (left) Visualization of the subject as a "stickman" figure using the mocap data. The layout of the room was approximated by looking at where different activities were performed. (right) Using the visualization to determine which index in the data-file corresponds to which body part.

was considered the first segment in the current sequence, and vice versa for the last segment (one sequence being the preparation of one recipe).

After reordering, we also re-labeled all the segments, with the goal of precisely determining the start and end of each micro-activity. To do that, we used the mocap data to visualize the marker positions in 3D space. The visualization was done using Unity [8] (Figure 2). We then used the same program to create a simple labeling tool, where we could watch the motion of a subject and try to visually infer their activity.

During the re-labeling process we created two sets of labels, so that each frame had two different labels attached to it. For the first set (*base* labeling), we were using only the labels of the challenge. For the second set (*precise* labeling), we additionally used the label *Undefined* if the activity performed was not one the organizers required us to recognize. For instance, for the cutting motion, we labeled as *Cut* the frames when the subject was cutting food, and as *Undefined* all the frames when they were doing something else (taking the knife, putting it on the table, etc.). This allowed us to later train and recognize the very specific motions of each micro-activity with a high accuracy.

3 Feature extraction

Using the raw mocap data, we first used the existing sensor streams (e.g, each mocap sensor position) to create additional, derived, sensor streams. These streams included the speed of movement, acceleration, and the distances and angles between some selected joints (e.g, distances between hands, elbows and shoulders, the distance between ankles, the angles between different hand parts, the distance to the floor etc.). The latter were chosen based on the expert knowledge acquired by looking at the visualizations and determining the relevant specifics of motion. After this procedure we had 129 sensor streams.

All the sensor streams (base and derived) were split into two-second windows, and from each window features were calculated. Larger windows were found impractical as they did not capture short micro-activities and micro-activities that started/ended at the border of the 30-second segments. The features include basic ones such as mean, variance, standard deviation, minimum, maximum, lower and upper quartiles etc., but also features computed by Fast Fourier Transform and some other features frequently used in similar domains (e.g., count above and below mean, and absolute sum of change).

While we also calculated features from the acceleration data that have proven themselves in our previous work [10], we found that including them does not increase the classification accuracy. We believe this is due to the variable sampling rate and missing values in the acceleration data, especially compared to the high-quality mocap data. In addition, since data from the accelerometer is missing a large proportion of the time, we had to make two classifiers for each task (one that used both sensor modalities, and one that only used mocap data). For these two reasons, we decided against using acceleration data in our final submission, and it will be thus omitted in the rest of the paper.

For the feature selection step we used a simple approach of ordering the features by the mutual information between the feature and the label. Then we trained models using the best n features, where n was a variable cut-off.

All the features described so far were used for the recognition of micro-activities. In order to recognize the recipe, we created another set of features. Since the re-ordering assembled all parts of each recipe into one sequence, we could use this full sequence as one instance. We used the general micro-activity recognizer (trained on base labels for all three recipes) on each sequence, and then computed the proportions of micro-activities in the first eighth of the sequence, second eighth and so on. Only the most well recognized micro-activities were used: *Take*, *Wash*, *Put* and *Cut*. These proportions alongside the length of the sequence became the features for the recipe recognition.

4 Classification

4.1 Classifiers

As described in Section 1.1, we started by using one general micro-activity classifier to classify all two-second windows. Next, we collected all the classifications for one sequence and from them calculated the features for that sequence – which we then classified into recipe. Depending on which recipe was detected, two recipe-specific micro-activity classifiers were used to classify the same two-second windows as before (hopefully, more accurately than with the general micro-activity classifier).

The reasoning for having two micro-activity classifiers is the following: as explained in Section 2.1 many of the actions performed by the recorded subjects did not fit into any of the available micro-activities, and the labels for those actions were essentially noise. We feared that as a consequence, some micro-activities would be hard to learn. To mitigate this, we had another set of labels (*precise*) that labeled all those actions as *Undefined*. One classifier was trained on the original labels (*base*) and the other on the *precise* labels. When classifying, the *precise* classifier made the predictions first. If the prediction was *Undefined*, the second classifier made another prediction to substitute it.

This whole pipeline therefore has 7 (one general, and two specialized for each recipe) different micro-activity classifiers and one recipe classifier. The recipe classifier was a simple Random Forest with default parameters. For all the 7 micro-activity classifiers we decided to use the same parameters: the same features, the same classifier type (Random Forest) and the same number of estimators in the Random Forest. The only difference between them was the data used for training (all data or only data from one recipe) and the type of labels used (*base* or *precise*).

The classifier and its hyper-parameters were selected empirically, as shown in Section 5. We tested different classical classifiers, and a custom deep learning network. For that, a deep Multi Task Learning (MTL) architecture was utilized, where each micro-activity was being represented as a separate task (one vs. all). The architecture had two fully connected layers shared across all the tasks and one task-specific layer. The final output of the model was provided by concatenating the outputs of the task-specific layers.

4.2 Post-processing with a Hidden Markov Model

Using only classical classification, all the windows are classified independently from one another. This approach discards all the information on temporal dependencies between them. If a subject is currently taking food from the cupboard (*Take*), for example, but the next window is *Cut*, followed by another *Take* classification, it is far more likely for *Cut* to be a misclassification than a micro-activity switch. In addition, the order of the micro-activities is more or less fixed, so if *Mix* is classified before *Pour* we can be certain that the recognition is wrong and that their order must be changed.

This motivated us to use an extra step after each classification, where the temporal information was taken into account using a HMM model. In this model we assume that we are moving through a number of hidden states, generating stochastic, but visible, emissions. In this case the hidden states represented the actual micro-activity, while the observed emissions represented the classified micro-activities. The parameters of this model are the transition probabilities between the states (transition matrix) and the probabilities of the observed emissions in each state (essentially a normalized confusion matrix).

The input to the model is an entire classified sequence (one run of the recipe from start to finish). This observed sequence could be generated by many different sequences of actual micro-activities, but HMM can determine the most likely one of them and return it as output. This output was our submission to the competition.

5 Results

First, we tested which machine-learning algorithm is the most suited for the micro-activity recognition task. We took several classic machine-learning algorithms from the *sklearn* library [9] in addition to the Extreme Gradient Boost (XGB) algorithm [11] and deep learning (Section 4.1). For this experiment we took the *base* labeling, split into two-second windows, and data from all recipes. We used the leave-one-subject-out scheme, where data from two subjects was used for training and data from the remaining one for testing. All the reported results in this section are the average from all three runs. Additionally, we chose to report the results using accuracy, as it is the most common and well known metric. However, for the final result we also give the score as defined by the competition. This score averages the mean of the accuracy of recipe classification and the mean of the accuracy of micro-activity classification.

From the results in Table 1 we can see that Random Forest was the most accurate one, surpassing both the deep learning approach and somewhat surprisingly the ensemble of all approaches (implemented by the majority vote).

Algorithm	Accuracy [%]	Algorithm	Accuracy [%]
Decision tree	52.7	k-NN	52.0
Bagging	61.7	SVM	47.3
Gradient boosting	55.0	XGB	62.0
Deep learning	65.7	MLP	52.0
Random Forest	68.0	Ensamble	67.7

Table 1 Accuracy for different micro-activity classifiers.

Having decided on the Random Forest classifier, we tested the impact of two parameters: the number of trees in the forest and the number of available features. For the number of trees, we sampled numbers between 1 and 2000, and observed that accuracy is slowly increasing up to around 1000 trees. Some sample results from this test are shown in Table 2. The feature selection process was described in Section 3 and some sample results, using different number of features, are again found in Table 2. They show, interestingly, that using all features gives us better performance than any tested subset.

The outputs from this micro-activity classifier were then used as features for the recipe classifier – implemented as a Random Forest. Each individual run of each recipe was one instance. We achieved 100% accuracy for this task, so for the further steps we could assume we always knew to which recipe any time window belongs.

The next step was to train two classifiers for each recipe as described in Section 4. We used the Random Forest classifier with the same parameters and features as in the first general micro-activity classifier. A case could be made for using different parameters or even classifiers for each recipe, but we chose the simpler approach to avoid overfitting.

The results for the *base* classifier for each recipe are shown in Table 5 and show that the accuracy increases substantially (from 69% to 80%) if classifiers are specialized.

# Trees	Accuracy [%]	# Features	Accuracy [%]
50 Trees	67.7	10 Features	65.4
100 Trees	68.0	100 Features	67.8
500 Trees	68.9	1000 Features	67.7
1000 Trees	69.0	1994 Features	69.0

Table 2 Accuracy [%] for different number of trees in the Random Forest and for different number of features used. When testing different number of trees, all features were used. Conversely, when testing different number of features, the maximum (1000) number of trees was used.

Using *precise* labels, the accuracy is even higher, but those labels contain *Undefined* activity that appears roughly 35% of the time, making the problem easier. When combining the predictions, the overall accuracy increases, but only by a negligible amount. We decided to still use this combination for our final competition predictions as it increased the recall of short micro-activities, but it is possible that this was not a crucial step in the pipeline.

Finally, we used HMM smoothing as described in Section 4.2. This again substantially boosted the results as can be seen in Table 4. Both the competition score for micro-activities as well as accuracy are around 90%. When combining this score with the recipe detection accuracy, we achieved the competition score of 95%.

	Sandwich	Cereal	Fruit salad	All
Base	76.7	80.3	82.3	79.8
Precise	83.6	90.0	82.0	85.2
Combined	76.0	81.3	83.3	80.2

Table 3 Accuracy [%] when using specialized classifiers for each recipe. Table shows the results for both sets of labels and their combination.

	Sandwich	Cereal	Fruit salad	All
Accuracy [%]	92.3	92.3	90.0	91.5
Competition score [%]	83.0	94.7	93.7	90.4

Table 4 Accuracy and the competition score after using HMM to smooth out the predictions.

6 Discussion

We believe that the high accuracy achieved by our approach stems from three main advantages. The first is that re-ordering and re-labeling the data adds a lot of temporal information to the dataset – both on the order of 30-second segments and on the exact timing of the micro-activities in them. This allowed us to use conventional machine learning techniques, which turned out to be completely adequate for this research problem. Another advantage is that we interleaved the recognition of micro-activities and recipes: the first helped us to determine the second, and when recipe was recognized it helped us to refine the micro-activity recognition. To do so we had to use multiple machine learning models, but in return we significantly improved our results. Finally, the last big advantage of our approach is the use of the HMM. As all subjects follow the same cooking procedure, the challenge data was very consistent and predictable – increasing the HMM effectiveness.

However, we can also point out some negative aspects of our approach. Firstly, while manual data labeling adds information to the dataset, it is very time-consuming. Finding a way to automate labeling could improve the possibilities of processing a larger amount of data, and thus obtain better results. Secondly, we chose to use the same model (Random Forest) for all our classifiers. This could be sub-optimal, as different models and/or parameters for each step of the classification could lead to more accurate results. Lastly, we chose not to use acceleration data at all as it did not increase the accuracy of our models – however finding a use for it, despite its inconsistencies, could be an interesting research problem.

We also wanted to mention that the competition score felt very "noisy" at times, and perhaps a different metric could be used in the future. The problem is that it is really difficult to determine the exact start and end of each micro-activity period. If a micro-activity started at the very end of a segment (or ended at the beginning of one) it would not be recognized for that segment, resulting in massive drop in accuracy for that segment even if most of it was correctly classified.

While the competition was challenging and interesting, it did not reflect well the challenges one would encounter when developing a system to support cooking in real life. The first issue was the dataset, which made the machine learning task artificially more difficult. The second issue are the sensors: mocap data was of very high quality, completely outclassing accelerometers, which are otherwise more realistic sensors in ambient assisted living. The third issue was the regularity of the preparation procedures, which made the learning easier than it would be on more naturalistic data. It would be interesting to repeat the competition next year with a more realistic dataset.

7 Conclusion

This paper describes our approach to the problem presented by the Cooking Activity Recognition Challenge. We identified three main problems with this task. The first was the unknown ordering of the data, the second was the type of labels given by the organizers and the final one the micro-activities themselves – some were very similar to each other, and some were very short and thus hard to detect.

We solved the first problem by reordering the data, which allowed us to employ the HMM model which substantially boosted our results (roughly 10 percentage points). The second problem was solved by re-labeling the whole training dataset by hand, which enabled the use of conventional machine learning techniques.

Our approach for the machine learning itself (solving the third problem) was fairly conventional, using a well tuned Random Forest. Nonetheless, we list some key insights from the tuning process: we used small window size (2 seconds) in order to better detect short activities, we trained specialized classifiers for each recipe which improved the accuracy by roughly 10% and we only used the mocap data – as the acceleration data was too inconsistent.

Our final approach had the competition score of 95% when doing cross-validation on the training set. Given that the test set did not exhibit any major statistical difference to the training one, we hope that our final submission will receive a similar score.

8 Appendix: General information

Sensor modalities	Mocap data
Features used	Section 3
Programming language	Python
Library used	sklearn, xgb, hmm, tsfresh
Window size	2 second
Post-processing	Hidden Markov Model
Training/testing time	< 1 minute
Hardware	Intel i7-4790, 16GB RAM

Table 5 General information for the used approach.

References

1. Brown RT, Komaiko KD, Shi Y, Fung KZ, Boscardin WJ, Au-Yeung A, Tarasovsky G, Jacob R, Steinman MA (2017). Bringing functional status into a big data world: Validation of national Veterans Affairs functional status data. Plos One, doi: 10.1371/journal.pone.0178726
2. Shen J, Naeim A (2017). Telehealth in older adults with cancer in the United States: The emerging use of wearable sensors. Journal of Geriatric Oncology, doi: 10.1016/j.jgo.2017.08.008
3. Sayeda Shamma Alia, Paula Lago, Shingo Takeda, Kohei Adachi, Brahim Benaissa, Md Atiqur Rahman Ahad and Sozo Inoue, 'Summary of the Cooking Activity Recognition Challenge', Human Activity Recognition Challenge, Smart Innovation, Systems and Technologies, Springer Nature, 2020
4. Paula Lago, Shingo Takeda, Kohei Adachi, Sayeda Shamma Alia, Moe Matsuki, Brahim Benaissa, Sozo Inoue and François Charpillet, 'Cooking Activity Dataset with Macro and Micro Activities', IEEE DataPort, doi: 10.21227/hyzg-9m49, 2020
5. Paula Lago, Shingo Takeda, Sayeda Shamma Alia, Kohei Adachi, Brahim Benaissa, François Charpillet and Sozo Inoue, 'A Dataset for Complex Activity Recognition with Micro and Macro Activities in a Cooking Scenario', preprint, 2020
6. Kubota A, Iqbal T, Shah JA, Riek LD (2019) Activity recognition in manufacturing: The roles of motion capture and sEMG+inertial wearables in detecting fine vs. gross motion. In: 2019 International Conference on Robotics and Automation (ICRA). Montreal, Canada
7. Mobark M, Chuprat S, Mantoro T (2017) Improving the accuracy of complex activities recognition using accelerometer-embedded mobile phone classifiers. In: 2017 Second International Conference on Informatics and Computing (ICIC). Jayapura, Indonesia
8. Unity https://unity.com/
9. scikit-learn https://scikit-learn.org/stable/
10. Cvetković, B., Szeklicki, R., Janko, V., Lutomski, P., Luštrek, M. (2018). Real-time activity monitoring with a wristband and a smartphone. Information Fusion, 43, 77-93.
11. XGB https://xgboost.readthedocs.io/