# Learning Deep Representations for Video-Based Intake Gesture Detection

Philipp V. Rouast, *Student Member, IEEE*, and Marc T. P. Adam

***Abstract*—Automatic detection of individual intake gestures during eating occasions has the potential to improve dietary monitoring and support dietary recommendations. Existing studies typically make use of on-body solutions such as inertial and audio sensors, while video is used as ground truth. Intake gesture detection directly based on video has rarely been attempted. In this study, we address this gap and show that deep learning architectures can successfully be applied to the problem of video-based detection of intake gestures. For this purpose, we collect and label video data of eating occasions using 360-degree video of 102 participants. Applying state-of-the-art approaches from video action recognition, our results show that (1) the best model achieves an $F_1$ score of 0.858, (2) appearance features contribute more than motion features, and (3) temporal context in form of multiple video frames is essential for top model performance.**

***Index Terms*—Deep learning, intake gesture detection, dietary monitoring, video-based.**

## I. Introduction

DIETARY monitoring plays an important role in assessing an individual's overall dietary intake and, based on this, providing targeted dietary recommendations. Dietitians [1] and personal monitoring solutions [2] rely on accurate dietary information to support individuals in meeting their health goals. For instance, research has shown that the global risk and burden of non-communicable disease is associated with poor diet and hence requires targeted interventions [3]. However, manually assessing dietary intake often involves considerable processing time and is subject to human error [4].

Automatic dietary monitoring aims to detect (i) *when*, (ii) *what*, and (iii) *how much* is consumed [5]. This is a complex and multi-faceted problem involving tasks such as action detection to identify intake gestures (*when*), object recognition and segmentation to identify individual foods (*what*), as well as volume and density estimation to derive food quantity (*how much*). A variety of sensors have been explored in the literature, including inertial, audio, visual, and piezoelectric sensors [5]–[7].

Detection of individual intake gestures can improve detection of entire eating occasions [8] and amounts consumed [9]. It also provides access to measures such as intake speed, as well as meta-information for easier review of videos. Although video is often used as ground truth for studies focused on detecting chews, swallows, and intake gestures, it has rarely been used as the basis for automatic detection. However, there are several indications that video could be a suitable data source to monitor such events: (i) increasing exploration of video monitoring in residential and hospital settings [10], [11], (ii) the rich amount of information embedded in the visual modality, and (iii) recent advances in machine learning, and in particular deep learning [12], for video action recognition that have largely been left unexplored in dietary monitoring.

In this paper, we address this gap by demonstrating the feasibility of using deep neural networks (DNNs) for automatic detection of intake gestures from raw video frames. For this purpose, we investigate the 3D CNN [13], CNN-LSTM [14], Two-Stream [15], and SlowFast [16] architectures which have been applied in the field of video action recognition, but not for dietary monitoring. These architectures allow to consider *temporal context* in the form of multiple frames. Further, instead of relying on handcrafted models and features, deep learning leverages a large number of examples to learn feature representations on multiple levels of abstraction. In dietary monitoring, deep learning has mainly been used for image-based food recognition (*what*) [17], and recently in intake gesture detection based on inertial sensors (*when*) [6]. However, it has yet to be applied on video-based intake gesture detection. Our main contributions are the following:

1) We fill the gap between dietary monitoring and video action recognition by demonstrating the feasibility of using deep learning architectures to detect individual intake gestures from raw video frames. We conduct a laboratory study with 102 participants and 4891 intake gestures, by sourcing video from a 360-degree camera placed in the center of the table. A ResNet-50 SlowFast model achieved the best $F_1$ score of 0.858.
2) Video action recognition can build on both appearance and motion features. It is in general not clear which are

Manuscript received May 16, 2019; revised August 15, 2019 and September 16, 2019; accepted September 18, 2019. Date of publication September 30, 2019; date of current version June 5, 2020.This work was supported in part by Bill and Melinda Gates Foundation under Grant OPP1171389 and in part by Australian Government Research Training (RTP) Scholarship. *(Corresponding author: Marc Adam.)*

The authors are with the School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW 2308, Australia (e-mail: philipp.rouast@uon.edu.au; marc.adam@newcastle.edu.au).

This article has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors. This contains static multi-frame versions as well as video versions of Fig. 6, 8, and 9 of the main article. This material is 582 kilobytes in size.

Digital Object Identifier 10.1109/JBHI.2019.2942845

more important for a given action [18]. Using a 2D CNN without temporal context, we show that appearance (individual frames) performs better than motion (optical flow between adjacent frames) for detecting intake gestures.

3) Similarly, it is not clear to what extent temporal context improves model accuracy of detecting a given action [19]. Comparing the best model *with* (ResNet-50 SlowFast) and the best 2D CNN *without* temporal context, we find a relative $F_1$ improvement of 8%.

The remainder of the paper is organized as follows: In Section II, we discuss the related literature, including dietary monitoring and video action recognition. Our proposed models are introduced in Section III, and the dataset in Section IV. We present our experiments and results in Section V, and draw conclusions in Section VI.

## II. Related Research

### A. Dietary Monitoring

At the conceptual level, dietary monitoring broadly captures three components of recognition, namely *what* (e.g., identification of specific foods), *how much* (i.e., quantification of consumed food), and *when* (i.e., timing of eating occasions). Traditional paper-based methods such as recalls and specialized questionnaires [20] are still commonly used by dietitians. Amongst end-users, mobile applications that allow manual logging of individual meals are also popular. These active methods are characterized by a considerable amount of effort, and known to be affected by biases and human error [4].

Realizing the requirement for objective measurements of a person's diet, several sensor-based approaches of passively collecting information associated with diet have been proposed in the literature. With the emergence of labeled databases of food images [17], [21], food recognition from still images has become a popular task in computer vision research. The state of the art uses features learned by deep convolutional neural networks (CNNs) to distinguish between food classes [22]. CNNs are DNNs especially designed for visual inputs.

Image-based estimation of food volume and associated calories typically extends food recognition by volume estimation of different foods, and linking with nutrient databases [23], [24]. Estimation of food volume from audio and inertial sensors based on individual bite sizes has also been proposed [9].

In detecting intake behavior, we distinguish between detection of events describing meal microstructure (e.g., individual intake gestures), and detecting intake occasions as a whole (e.g., a meal), which can be seen as clusters of detected events [25]. Besides aiding in the estimation of food volume [9], information about meal microstructure can be leveraged to improve active methods [26]. It also allows dietitians to quantify measures of interest such as the rate of eating [27].

In general, detection of chews and swallows is typically attempted using on-body audio or piezoelectric sensors, whilst detection of intake gestures is the domain of wrist-mounted inertial sensors [28]. Chews and swallows generate characteristic audio signatures, which was exploited for automatic detection of meal microstructure as early as 2005 [29], [30]. Swallows can also be registered using piezoelectric sensors measuring strain on the jaw [31]. Inertial sensors can be used to measure the acceleration and spatial movements of the wrist to identify intake gestures [32]–[34]. Recently, DNNs were applied for this purpose [6].

### B. Video-Based Intake Gesture Recognition

Despite the importance of visual sensors for recording ground truth, video data of eating occasions is rarely considered as the basis for automatic detection of meal microstructure. This is surprising, as the visual modality contains a broad range of information about intake behavior. In fact, in 2004, one of the earliest works in this field considered surveillance type video recorded in a nursing home to detect intake gestures [35]. This approach relied on optical flow-based motion features, which were used to train a Hidden Markov Model. A further approach used object detection of face, mouth, and eating utensils which was realised with haar-like appearance features [36]. We also see skeleton-based approaches with additional depth information [37], [38]. Deep learning, which is the state of the art for video action recognition, has not been explored to the best of our knowledge.

### C. Video Action Recognition

The task of action recognition from video extends 2D image input by the dimension of time. While temporal context can carry useful information, it also complicates the search for good feature representations given the typically much larger dimensionality of the raw data. Before the proliferation of deep learning, approaches in video action recognition would follow the traditional paradigm of pattern recognition: Computing complex hand-crafted features from raw video frames, based on which shallow classifiers could be learned. Such features were either video-level aggregation of local spatio-temporal features such as HOG3D [39], or point trajectories of dense points computed, e.g., using optical flow [40]. The following four deep learning architectures emerged from the literature on video action recognition, as shown in Fig. 1:

*1) 3D CNN – Spatio-Temporal Convolutions:* The 3D CNN approach features 3D convolutions instead of the 2D convolutions found in 2D CNNs. Videos are treated as spatio-temporal volumes, where the third dimension represents temporal context. 3D CNNs can thus automatically learn low-level features that take into account both spatial and temporal information. This approach was first proposed in 2010 by Ji *et al.* [13], who integrated 3D convolutions with handcrafted features. Running experiments with end-to-end training on larger datasets, Karpathy *et al.* [19] reported that it works best to slowly fuse the temporal information throughout the network. However, they found that temporal context only improves model accuracy for some classes such as juggling; furthermore, it reduced accuracy for some classes [19]. Other experiments regarding architecture choices concluded that 3D CNN can model appearance and motion simultaneously [42].

*2) CNN-LSTM – Incorporating Recurrent Neural Networks:* In the CNN-LSTM approach, the temporal context is modelled

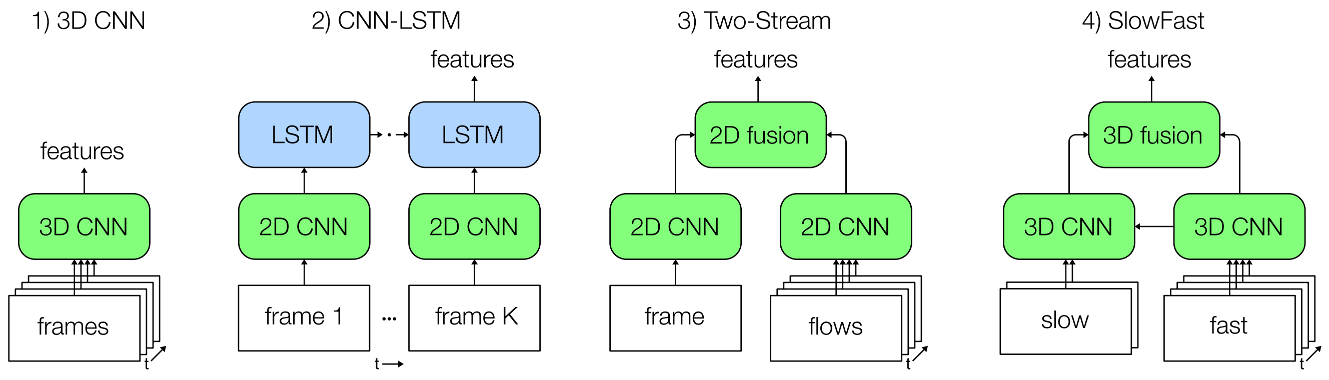


Fig. 1. The four investigated approaches from video action recognition based on temporal context $t$ (adapted from Carreira and Zisserman [41]).

by a recurrent neural network (RNN). RNNs are DNNs that take the previous model state as an additional input. In 2015, Donahue *et al.* [14] proposed to use the sequence of high-level spatial features learned by a CNN from individual video frames as input into a long short-term memory (LSTM) RNN. Such LSTM networks are known to be easier to train for longer sequences [43]. The CNN-LSTM model has the advantage of being more flexible with regards to the number of input frames, but has relatively many parameters and appears to be more data hungry in comparison to other approaches [41].

*3) Two-Stream – Decoupling Appearance and Motion:* In 2014, Simonyan and Zisserman [15] observed that 2D CNN models without temporal context achieved accuracy close to the 3D CNN approach [19], and that state-of-the-art accuracies involved handcrafted trajectory-based representations based on motion features. They proposed the two-stream architecture, which decouples appearance and motion by using a single still frame (appearance) and temporal context in form of stacked optical flow (motion). Both are fed into separate CNNs, where the appearance CNN is pre-trained on the large ImageNet database. While the original design employed score-level fusion [15], later variants used feature-level fusion of the last CNN layers [44].

*4) SlowFast – Joint Learning at Different Temporal Resolutions:* The SlowFast architecture proposed by Feichtenhofer *et al.* [16] in late 2018 learns from temporal context at multiple temporal resolutions. As of mid 2019, it represents the state of the art in video action recognition with 79% accuracy on the large Kinetics dataset without any pre-training. The idea of decoupling slow and fast motion is integrated into the network design. Two pathways make up the SlowFast architecture, consisting of a 3D CNN each: The *slow* pathway has more capacity to learn about appearance than motion, while the *fast* pathway works the other way around. This is realized by setting a factor $\alpha$ as the difference in sequence downsampling, and a factor $\beta$ as the difference in learned channels. A number of lateral connections allow that information from both pathways is fused.

## III. Proposed Method

Detecting individual intake gestures from video requires prediction of sparse points in time. We adopt the approach of

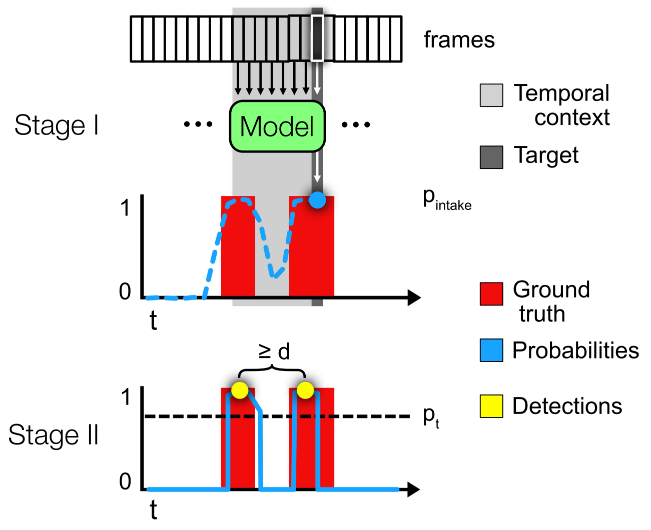


Fig. 2. Illustration of sample outputs at the two stages. In stage I, the model estimates the probability $p_{intake}$ for a target frame. For models with temporal context, input consists of multiple frames (16 in our experiments), of which the last frame is the target. In stage II, detections of intake events are realized using a local maximum search on the $p_t$-thresholded series of probabilities, where the detections have to be at least $d$ apart (in our experiments, $d = 2$ $s$).

Kyritsis *et al.* [6] and split this problem into two stages, as illustrated in Fig. 2:

*Stage I:* **Estimation of state probability at the frame level**, i.e., estimating the probability $p_{intake}$ for each frame, and

*Stage II:* **Detection of intake gestures**, by selecting sparse points in time based on the estimated probabilities.

### A. Stage I: Models for Frame-Level Probability Estimation

In Stage I, our models estimate $p_{intake}$, which is the probability that the label of the target frame is "intake". The four models identified from the literature on video action recognition represent our main models (3D CNN, CNN-LSTM, Two-Stream, SlowFast; see Fig. 1). In addition to the target frame, each of these four models considers a temporal context of further frames

TABLE I

**Small Model Instantiations.** We Report Temporal, Spatial, and Channel Dim. of Convolution Kernels as $\{T \times S^2, C\}$, Temporal and Spatial Dim. of Pooling Ops. as $\{T \times S^2\}$, and Stride Sizes Likewise. Corresponding Output Sizes are Reported as $\{T \times S^2 \times C\}$

| Layer | 0a) 2D CNN[a] | | 1a) 3D CNN | | 2a) CNN-LSTM | | 3a) Two-Stream[a] | | 4a) SlowFast[b] | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | frame<br>flow | | | | | | frame<br>flows | | slow<br>fast |
| data | | $128^2 \times 3\|2$ | | $16 \times 128^2 \times 3$ | | $16 \times 128^2 \times 3$ | | $128^2 \times 3$<br>$128^2 \times 32$ | | $4 \times 128^2 \times 3$<br>$16 \times 128^2 \times 3$ |
| conv0[c] | | | | | | | $3^2, 3$<br>str. $1^2$ | $128^2 \times 3$<br>$128^2 \times 3$ | | |
| conv1 | $3^2, 32$<br>str. $1^2$ | $128^2 \times 32$ | $3 \times 3^2, 32$<br>str. $1 \times 1^2$ | $16 \times 128^2 \times 32$ | $3^2, 32$<br>str. $1^2$ | $16 \times 128^2 \times 32$ | $3^2, 32$<br>str. $1^2$ | $128^2 \times 32$<br>$128^2 \times 32$ | $1\|3 \times 3^2, 32\|8$<br>str. $1 \times 1^2$ | $4 \times 128^2 \times 32$<br>$16 \times 128^2 \times 8$ |
| pool1 | $2^2$<br>str. $2^2$ | $64^2 \times 32$ | $2 \times 2^2$<br>str. $2 \times 2^2$ | $8 \times 64^2 \times 32$ | $2^2$<br>str. $2^2$ | $16 \times 64^2 \times 32$ | $2^2$<br>str. $2^2$ | $64^2 \times 32$<br>$64^2 \times 32$ | $1 \times 2^2$<br>str. $1 \times 2^2$ | $4 \times 64^2 \times 32$<br>$16 \times 64^2 \times 8$ |
| conv2 | $3^2, 32$<br>str. $1^2$ | $64^2 \times 32$ | $3 \times 3^2, 32$<br>str. $1 \times 1^2$ | $8 \times 64^2 \times 32$ | $3^2, 32$<br>str. $1^2$ | $16 \times 64^2 \times 32$ | $3^2, 32$<br>str. $1^2$ | $64^2 \times 32$<br>$64^2 \times 32$ | $1\|3 \times 3^2, 32\|8$<br>str. $1 \times 1^2$ | $4 \times 64^2 \times 32$<br>$16 \times 64^2 \times 8$ |
| pool2 | $2^2$<br>str. $2^2$ | $32^2 \times 32$ | $2 \times 2^2$<br>str. $2 \times 2^2$ | $4 \times 32^2 \times 32$ | $2^2$<br>str. $2^2$ | $16 \times 32^2 \times 32$ | $2^2$<br>str. $2^2$ | $32^2 \times 32$<br>$32^2 \times 32$ | $1 \times 2^2$<br>str. $1 \times 2^2$ | $4 \times 32^2 \times 32$<br>$16 \times 32^2 \times 8$ |
| conv3 | $3^2, 64$<br>str. $1^2$ | $32^2 \times 64$ | $3 \times 3^2, 64$<br>str. $1 \times 1^2$ | $4 \times 32^2 \times 64$ | $3^2, 64$<br>str. $1^2$ | $16 \times 32^2 \times 64$ | $3^2, 64$<br>str. $1^2$ | $32^2 \times 64$<br>$32^2 \times 64$ | $1\|3 \times 3^2, 64\|16$<br>str. $1 \times 1^2$ | $4 \times 32^2 \times 64$<br>$16 \times 32^2 \times 16$ |
| pool3 | $2^2$<br>str. $2^2$ | $16^2 \times 64$ | $2 \times 2^2$<br>str. $2 \times 2^2$ | $2 \times 16^2 \times 64$ | $2^2$<br>str. $2^2$ | $16 \times 16^2 \times 64$ | $2^2$<br>str. $2^2$ | $16^2 \times 64$<br>$16^2 \times 64$ | $1 \times 2^2$<br>str. $1 \times 2^2$ | $4 \times 16^2 \times 64$<br>$16 \times 16^2 \times 16$ |
| conv4 | $3^2, 64$<br>str. $1^2$ | $16^2 \times 64$ | $3 \times 3^2, 64$<br>str. $1 \times 1^2$ | $2 \times 16^2 \times 64$ | $3^2, 64$<br>str. $1^2$ | $16 \times 16^2 \times 64$ | $3^2, 64$<br>str. $1^2$ | $16^2 \times 64$<br>$16^2 \times 64$ | $1\|3 \times 3^2, 64\|16$<br>str. $1 \times 1^2$ | $4 \times 16^2 \times 64$<br>$16 \times 16^2 \times 16$ |
| pool4 | $2^2$<br>str. $2^2$ | $8^2 \times 64$ | $2 \times 2^2$<br>str. $2 \times 2^2$ | $1 \times 8^2 \times 64$ | $2^2$<br>str. $2^2$ | $16 \times 8^2 \times 64$ | $2^2$<br>str. $2^2$ | $8^2 \times 64$<br>$8^2 \times 64$ | $1 \times 2^2$<br>str. $1 \times 2^2$ | $4 \times 8^2 \times 64$<br>$16 \times 8^2 \times 16$ |
| fusion | | | | | | | | $8^2 \times 64$ | | $8^2 \times 64$ |
| flatten | | 4096 | | 4096 | | $16 \times 4096$ | | 4096 | | 4096 |
| dense | | 1024 | | 1024 | | $16 \times 1024$ | | 1024 | | 1024 |
| lstm | | | | | | $16 \times 128$ | | | | |
| dense | | 2 | | 2 | | $16 \times 2$ | | 2 | | 2 |

[a]For 2D CNN and Two-Stream, colors red and blue highlight how dimensions differ between frames and flows.
[b]For SlowFast, colors orange|cyan highlight the differences in model parameters and dimensions between the slow and fast pathways.
[c]Only for flow input; Serves the purpose of producing 3 channels for transfer learning.

preceding the target frame. As a baseline and for experiments, we additionally employ a 2D CNN. Because the 2D CNN does not have a temporal context, this enables us to (1) discern to what extent the temporal context improves model performance and (2) directly compare the importance of appearance and motion features.

For each model, we propose a small instantiation with relatively few parameters, and a larger instantiation using ResNet-50 [45] as backbone. In the following, we present each of the proposed models adapted for food intake gesture detection (see Tables I and II for details). Source code for all models is available at https://github.com/prouast/deep-intake-detection.

*0) 2D CNN:* The 2D CNN functions as a baseline for our study, indicating what is possible without temporal context. This allows us to discern the importance of the temporal context for intake gesture detection. Further, the 2D CNN also allows us to directly compare a model based solely on motion to one solely based on visual appearance. This assessment is not possible for the other four models.

Motion information can be of importance for classes with fast movement such as juggling [19]. For detection of intake gestures, it seems intuitive that appearance may be the more important modality, which is what we are seeking to confirm here. For appearance input is *the single target frame*, and for motion *the optical flow between the target frame and the frame directly preceding it*. We use Dual TV-L$^1$ optical flow [46], which produces two channels of optical flow corresponding to the horizontal and vertical components, as opposed to three RGB channels for frames.

0a) Small instantiation] A five-layer CNN of the architecture type popularised by AlexNet [47].

0b) ResNet-50 instantiation] We adopt the architecture given by [45], which allows us to use pre-trained models.

*1) 3D CNN:* This model has the ability to learn spatio-temporal features. We extend the 2D CNN introduced in the previous section by using 3D instead of 2D convolutions. The third dimension corresponds to the temporal context. We use temporal pooling following the slow fusion approach [19].

1a) Small instantiation] Extending the small 2D CNN to 3D, we use temporal convolution kernels of size 3 as recommended by [42]; temporal pooling is realized in the max pooling layers.

1b) ResNet-50 instantiation] We extend ResNet-50 [45] to 3D, but modify the dimensions to fit our input, since we do not use transfer learning for the 3D CNN. Within each block, the first convolutional layer has a temporal kernel size of 3, a choice adopted from [16]. Temporal fusion is facilitated by using temporal stride 2 in the second convolutional layer of the first block in each block layer.

TABLE II

**ResNet-50 Model Instantiations.** We Report Temporal, Spatial, and Channel Dim. of Conv. Kernels as $\{T \times S^2, C\}$, Temporal and Spatial Dim. of Pooling Ops. as $\{T \times S^2\}$, and Stride Sizes Likewise. The Corresponding Output Sizes Are Reported as $\{T \times S^2 \times C\}$

| Layer | 0b) 2D CNN[a] | | 1b) 3D CNN | | 2b) CNN-LSTM | | 3b) Two-Stream[a] | | 4b) SlowFast[b] | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | frame flow | | | | | | frame flows | | slow fast |
| data | | $224^2$ $\times 3 \mid 2$ | | $16 \times 128^2$ $\times 3$ | | $16 \times 224^2$ $\times 3$ | | $224^2 \times 3$ $224^2 \times 32$ | | $2 \times 128^2 \times 3$ $16 \times 128^2 \times 3$ |
| conv0[c] | $3^2, 3$ stride $1^2$ | $112^2$ $\times 3$ | | | | | $3^2, 3$ stride $1^2$ | $224^2 \times 3$ $224^2 \times 3$ | | |
| conv1 | $7^2, 64$ stride $2^2$ | $112^2$ $\times 64$ | $3 \times 5^2, 64$ stride $1 \times 1^2$ | $16 \times 128^2$ $\times 64$ | $7^2, 64$ stride $2^2$ | $16 \times 112^2$ $\times 64$ | $7^2, 64$ stride $2^2$ | $112^2 \times 64$ $112^2 \times 64$ | $1 \mid 3 \times 5^2, 64 \mid 8$ stride $1 \times 1^2$ | $2 \times 128^2 \times 64$ $16 \times 128^2 \times 8$ |
| pool1 | $3^2$ stride $2^2$ | $56^2$ $\times 64$ | $3 \times 3^2$ stride $2 \times 2^2$ | $8 \times 64^2$ $\times 64$ | $3^2$ stride $2^2$ | $16 \times 56^2$ $\times 64$ | $3^2$ stride $2^2$ | $56^2 \times 64$ $56^2 \times 64$ | $1 \times 3^2$ stride $1 \times 2^2$ | $2 \times 64^2 \times 64$ $16 \times 64^2 \times 8$ |
| res2.x ($\times 3$) | $1^2, 64$ $3^2, 64$ $1^2, 256$ | $56^2$ $\times 256$ | $3 \times 1^2, 64$ $1 \times 3^2, 64$ $1 \times 1^2, 256$ | $8 \times 64^2$ $\times 256$ | $1^2, 64$ $3^2, 64$ $1^2, 256$ | $16 \times 56^2$ $\times 256$ | $1^2, 64$ $3^2, 64$ $1^2, 256$ | $56^2 \times 256$ $56^2 \times 256$ | $1 \mid 3 \times 1^2, 64 \mid 8$ $1 \times 3^2, 64 \mid 8$ $1 \times 1^2, 256 \mid 32$ | $2 \times 64^2 \times 256$ $16 \times 64^2 \times 32$ |
| res3.x ($\times 4$) | $1^2, 128$ $3^2, 128$ $1^2, 512$ | $28^2$ $\times 512$ | $3 \times 1^2, 128$ $1 \times 3^2, 128$ $1 \times 1^2, 512$ | $4 \times 32^2$ $\times 512$ | $1^2, 128$ $3^2, 128$ $1^2, 512$ | $16 \times 28^2$ $\times 512$ | $1^2, 128$ $3^2, 128$ $1^2, 512$ | $28^2 \times 512$ $28^2 \times 512$ | $1 \mid 3 \times 1^2, 128 \mid 16$ $1 \times 3^2, 128 \mid 16$ $1 \times 1^2, 512 \mid 64$ | $2 \times 32^2 \times 512$ $16 \times 32^2 \times 64$ |
| res4.x ($\times 6$) | $1^2, 256$ $3^2, 256$ $1^2, 1024$ | $14^2$ $\times 1024$ | $3 \times 1^2, 256$ $1 \times 3^2, 256$ $1 \times 1^2, 1024$ | $2 \times 16^2$ $\times 1024$ | $1^2, 256$ $3^2, 256$ $1^2, 1024$ | $16 \times 14^2$ $\times 1024$ | $1^2, 256$ $3^2, 256$ $1^2, 1024$ | $14^2 \times 1024$ $14^2 \times 1024$ | $3 \times 1^2, 256 \mid 32$ $1 \times 3^2, 256 \mid 32$ $1 \times 1^2, 1024 \mid 128$ | $2 \times 16^2 \times 1024$ $16 \times 16^2 \times 128$ |
| res5.x ($\times 3$) | $1^2, 512$ $3^2, 512$ $1^2, 2048$ | $7^2$ $\times 2048$ | $3 \times 1^2, 512$ $1 \times 3^2, 512$ $1 \times 1^2, 2048$ | $1 \times 8^2$ $\times 2048$ | $1^2, 512$ $3^2, 512$ $1^2, 2048$ | $16 \times 7^2$ $\times 2048$ | $1^2, 512$ $3^2, 512$ $1^2, 2048$ | $7^2 \times 2048$ $7^2 \times 2048$ | $3 \times 1^2, 512 \mid 64$ $1 \times 3^2, 512 \mid 64$ $1 \times 1^2, 2048 \mid 256$ | $2 \times 8^2 \times 2048$ $16 \times 8^2 \times 256$ |
| fusion | | | | | | | | $7^2 \times 2048$ | | $1 \times 1^2 \times 2560$ |
| spatial pool | | $1^2 \times$ 2048 | | $1 \times 1^2 \times$ 2048 | | $16 \times 1^2 \times$ 2048 | | $1^2 \times 2048$ | | |
| flatten | | 2048 | | 2048 | | $16 \times 2048$ | | 2048 | | 2560 |
| lstm | | | | | | $16 \times 128$ | | | | |
| dense | | 2 | | 2 | | $16 \times 2$ | | 2 | | 2 |

[a]For 2D CNN and Two-Stream, colors red and blue highlight how dimensions differ between frames and flows.
[b]For SlowFast, colors orange|cyan highlight the differences in model parameters and dimensions between the slow and fast pathways.
[c]Only for flow input; Serves the purpose of producing 3 channels for transfer learning.

*2) CNN-LSTM:* The CNN-LSTM adds an LSTM layer to model a sequence of high-level features learned from raw frames. Note that this does not allow the model to learn low-level spatio-temporal features (as opposed to 3D CNN). Given the clear temporal structure of intake gestures (movement towards the mouth and back), it does however seem intuitive that knowledge of the development of high-level features from temporal context could help predict the current frame.

2a) Small instantiation] We use the features from the first dense layer of the small 2D CNN described previously as input into one LSTM layer with 128 units.

2b) ResNet-50 instantiation] The spatially pooled output of a ResNet-50's [45] last block is used as input into one LSTM layer with 128 units.

*3) Two-Stream:* For our instantiations of the Two-Stream approach, we follow the original work by Simonyan and Zisserman [15] to select the model input: The appearance stream takes the target frame as input; meanwhile, the motion stream is based on the stacked horizontal and vertical components of optical flow calculated using Dual TV-L[1] from pairs of consecutive frames in the temporal context.

3a) Small instantiation] Motion and appearance stream both follow the small 2D CNN architecture; after the last pooling layer, the streams are pooled using spatially aligned conv fusion as proposed by [44].

3b) ResNet-50 instantiation] Motion and appearance stream both follow the ResNet-50 [45] architecture; after the last block layer, the streams are pooled using spatially aligned conv fusion [44].

*4) SlowFast:* The SlowFast model processes the temporal context at two different temporal resolutions. Since our dataset has fewer frames than in the original work [16], we choose the factors $\alpha = 4$ and $\beta = 0.25$ for our SlowFast instantiations.

4a) Small instantiation] Both pathways are based on the small 2D CNN; we extend the convolutional layers to 3D and set the temporal kernel size to 1 for the slow pathway and to 3 for the fast pathway. Following [16], we choose time-strided convolutions of kernel size $3 \times 1^2$ for a lateral connection after each of the four convolutional layers. Fusion consists of temporal average pooling and spatially aligned 2D conv fusion [44].

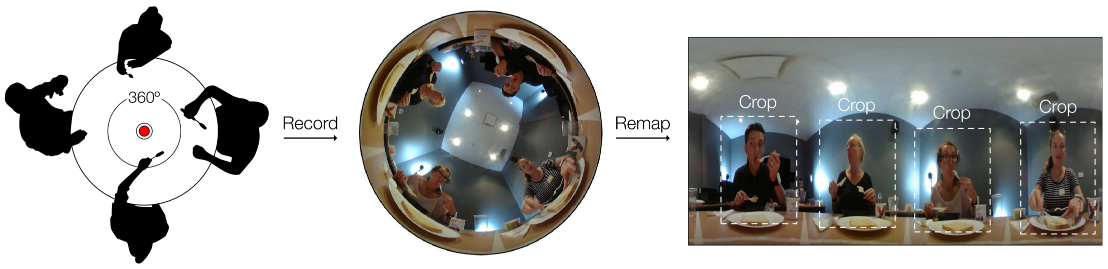


Fig. 3. Recording of one session. The spherical video is remapped to equirectangular representation, cropped, and reshaped to square shape.

4b) ResNet-50 instantiation] We directly follow [16] who themselves used ResNet-50 as backbone for SlowFast, only using the same dimension tweaks as in our ResNet-50 2D CNN. Fusion consists of global average pooling and concatenation.

## B. Loss Calculation

We use cross-entropy loss for all our models. At evaluation time, we only consider the target frame for prediction, which corresponds to the last frame of the input (see Fig. 2). The same applies to loss calculation during training for all models except CNN-LSTM: Following [41], we train the CNN-LSTM using the labels of all input frames, but evaluate only using the label of the target frame.

Due to the nature of our data, the classes are very imbalanced with many more "non-intake" frames than "intake" frames. When computing mini-batch loss, we correct for this imbalance by using weights calculated as

$$w_i = \frac{m}{C(i) * n} \tag{1}$$

to scale the loss for the $m$ labels $\mathbf{y} = \{y_1, \ldots, y_m\}$ in each minibatch, where $n$ is the number of classes and $C(i)$ is the number of elements of $\mathbf{y}$ which equal $y_i$.

## C. Stage II: Detecting Intake Gestures

We follow a maximum search approach [6] to determine the sparse individual intake gesture times. Based on estimated frame-level probabilities $\mathbf{p}$, we first derive $\mathbf{p}'$ by setting all probabilities below a threshold $p_t$ to zero. This leaves all frames $\{f : p_{intake,f} \geq p_t\}$ as candidates for detections, as seen at the bottom of Fig. 2. Subsequently, we perform a search for local maxima in $\mathbf{p}'$ with a minimum distance $d$ between maxima. The intake gesture times are then inferred from the indices of the maxima.

## IV. Dataset

We are not aware of any publicly available dataset including labeled video data of intake gestures. Related studies that involved collection of video data as ground truth typically do not make the video data available, and instead focus on the inertial [6] and audio sensor data [48].

For this research, we collected and labeled video data of 102 participants consuming a standardized meal of lasagna, bread, yogurt, and water in a group setting (ethics approval H-2017-0208). The data was collected in sessions of four participants at a time seated around a round table in a closed room without outside interference. Participants were invited to consume their meal in a natural way[1] and encouraged to have a conversation in the process. A 360fly-4 K camera was placed in the center of the table, recording all four participants simultaneously. As illustrated in Fig. 3, raw spherical video was first remapped to equirectangular representation. We then cropped out a separate video for each individual participant such that the dimensions include typical intake gestures. Each video was trimmed in time to only include the duration of the meal, and spatially scaled to a square shape.

Two independent annotators labeled and cross-checked the intake gestures in all 102 videos as durations using ChronoViz.[2] Each gesture is assigned as start timestamp the point where the final uninterrupted movement towards the mouth starts; as end timestamp, it is assigned the point when the participant has finished returning their hand(s) from the movement or started a different action. Based on the start and end timestamps, we derive a label for each video frame according to the following procedure: If a video frame was taken between start and end of an annotated gesture, it is assigned the label "intake". If a video frame is taken outside of any annotated gestures, it is assigned the label "non-intake". The dataset is available from the authors on request.

## V. Experiments

We use a global split of our dataset into 62 participants for training, 20 participants for validation, and 20 participants for test as summarized in Table III. To reduce computational burden, we downsample the video from 24 fps to 8 fps, and resize to dimensions 140 × 140 (128 × 128 after augmentation).

## A. Stage I: Estimating Frame-Level Intake Probability

We apply the models introduced in Section III to classify frames according to the two labels "intake" and "non-intake".

[1]After the meal, 64 of the 102 participants (63%) responded to the statement "The presence of the video camera changed my eating behavior" (5-point Likert scale, ranging from (1) strongly disagree to (5) strongly agree). With an average score of 2.11, we conclude that participants did not feel that the presence of the camera considerably affected their eating behavior.

[2]See http://chronoviz.com

TABLE III
SUMMARY STATISTICS FOR OUR DATASET AND THE TRAINING/VALIDATION/TEST SPLIT

| | Training | | | Validation | | | Test | | | Total | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Type | # | Mean [s] | Std [s] | # | Mean [s] | Std [s] | # | Mean [s] | Std [s] | # | Mean [s] | Std [s] |
| Participants | 62 | 802.25 | 243.31 | 20 | 785.97 | 243.74 | 20 | 891.00 | 222.27 | 102 | 816.46 | 240.07 |
| Intake Gestures | 2924 | 2.35 | 1.03 | 952 | 2.28 | 1.00 | 997 | 2.29 | 1.01 | 4891 | 2.32 | 1.02 |

For our experiments, we distinguish between models without and with temporal context:

- Models *without* temporal context (0a–0b) are of interest as a baseline, and to experimentally compare appearance and motion features. For appearance, input is the *single target frame*, and for motion, *optical flow between the target frame and the one preceding it*.
- For the models *with* temporal context (1a–4b), input consists of 16 frames, which corresponds to 2 seconds at 8 fps. The last of these frames is the prediction target. To take maximum advantage of the available training data, we generate input using a window shifting by one frame. The use of temporal context implies that the first 15 labels are not predicted.

*1) Training:* We use the Adam optimizer to train each model on the training set. Training runs for 60 epochs with a learning rate starting at 3e-4 and exponentially decaying at a rate of 0.9 per epoch. Models without temporal context are trained using batch size 64, while models with temporal context are trained using batch size 8.[3] Using the validation set, we keep track of the best models in terms of unweighted average recall (UAR), which is not biased by class imbalance. For regularization, we use l2 loss with a lambda of 1e-4. Dropout is used in all small instantiations of our models on convolutional and dense layers with rate 0.5, but we do not use dropout for the ResNet-50 instantiations. We also use data augmentation by dynamically applying random transformations: Small rotations, cropping to size 128 × 128, horizontal flipping, brightness and contrast changes. All models are learned end-to-end, optical flow is precomputed using Dual TV-L$^1$ [46].

*2) Transfer Learning and Warmstarting for Better Initial Parameters:* While the initial small 2D CNN is trained from scratch, we use it to warmstart the convolutional layers of both the small CNN-LSTM and the small Two-Stream model. The ResNet-50 2D CNN is initialized using an off-the-shelf ResNet-50 trained on the ImageNet database. To fit ImageNet dimensions, we resize our inputs for this model to 224 × 224, as listed in Table II. We use the ResNet-50 2D CNN to warmstart the convolutional layers of both the ResNet-50 CNN-LSTM and the ResNet-50 Two-Stream model. All 3D-CNN and SlowFast models are trained from scratch.

## B. Stage II: Detecting Intake Gestures

For the detection of intake gestures, we build on the exported frame-level probabilities using the models trained in Stage I.

[3] Batch sizes were chosen considering space constraints training on NVIDIA Tesla V100 at fp32 accuracy, and to be consistent across models.

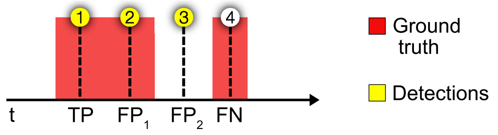


Fig. 4. The evaluation scheme proposed by Kyritsis *et al.* [6]. (1) A true positive is the first detection within each ground truth event; (2) False positives of type one are further detections within the same ground truth event; (3) False positives of type two are detections outside ground truth events; (4) False negatives are non-detected ground truth events.

We then apply the approach described in Section III-C to determine sparse detections.

*1) Evaluation Scheme:* We use the evaluation scheme proposed by Kyritsis *et al.* [6] as seen in Fig. 4. According to the scheme, one correct detection per ground truth event counts as a true positive ($TP$), while further detections within the same ground truth event are false positives of type 1 ($FP_1$). Detections outside ground truth events are false positives of type 2 ($FP_2$), and non-detected ground truth events count as false negatives ($FN$). Based on the aggregate counts, we calculate precision ($\frac{TP}{TP+FP_1+FP_2}$), recall ($\frac{TP}{TP+FN}$), and the $F_1$ score ($2 * \frac{Precision*Recall}{Precision+Recall}$).

*2) Parameter Setting:* The approach described in Section III-C requires setting two hyperparameters: The minimum distance between detections $d$, and the threshold $p_t$. We follow Kyritsis *et al.* [6] and set $d = 2\,s$, which approximates the mean duration of intake gestures, see Table III. Since we only run one final evaluation of each model on the test set, we use the validation set to approximate a good threshold $p_t$. Hence, for each model, we run a grid search between 0.5 and 1 on the validation set using a step size of 0.001 and choose the threshold that maximizes $F_1$. Table IV lists the final $\hat{\mathbf{p_t}}$.

## C. Results

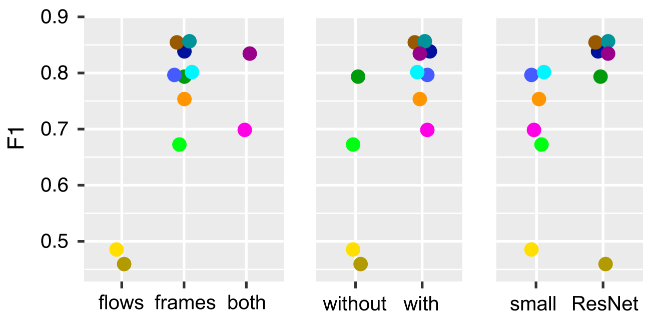
The best result is achieved by the state-of-the-art ResNet-50 SlowFast network with an $F_1$ score of 0.858. In general, we find that model accuracy is impacted by three factors of model choice, namely (i) frame or flow features, (ii) with or without temporal context, and (iii) model depth. Fig. 5 illustrates this by plotting the $F_1$ values grouped by each of these factors.

*1) Frame and Flow Features, Fig. 5(a):* Using the 2D CNN, we are able to directly compare how frame (appearance) and flow (motion) features affect model performance. For the small and ResNet instantiations, frame features lead to a relative $F_1$ improvement of 38% and 72% over flow features. An improvement is also measurable for UAR. Further, the Two-Stream models, which mainly rely on flow features, perform worse than

TABLE IV
RESULTS FOR STAGES I AND II. REPORTED VALUES ARE BASED ON THE TEST SET, WHICH WAS ONLY EVALUATED ONCE

| Model | Features used[a] | | Temporal context[b] | Stage I | | Stage II[c] | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Frames | Flows | | #Params | UAR | $\hat{p}_t$ | $TP$ | $FP_1$ | $FP_2$ | $FN$ | $F_1$ |
| 0a) 🟩 Small 2D CNN | ✓ | | Without | 4.26M | 82.63% | 0.957 | 670 | 39 | 287 | 321 | 0.674 |
| 0a) 🟨 Small 2D CNN | | ✓ | | 4.26M | 71.76% | 0.793 | 662 | 45 | 1023 | 329 | 0.487 |
| 0b) 🟩 ResNet-50 2D CNN | ✓ | | | 23.5M | 86.39% | 0.964 | 829 | 54 | 211 | 162 | 0.795 |
| 0b) 🟨 ResNet-50 2D CNN | | ✓ | | 23.5M | 71.34% | 0.865 | 661 | 53 | 1163 | 330 | 0.461 |
| 1a) 🟦 Small 3D CNN | ✓ | | With | 4.39M | 87.54% | 0.997 | 795 | 37 | 169 | 196 | 0.798 |
| 2a) 🟧 Small CNN-LSTM | ✓ | | | 4.85M | 83.36% | 0.983 | 674 | 17 | 104 | 317 | 0.755 |
| 3a) 🟪 Small Two-Stream | ✓ | ✓ | | 4.34M | 81.96% | 0.973 | 653 | 36 | 185 | 338 | 0.700 |
| 4a) 🟦 Small SlowFast | ✓ | | | 4.49M | 88.71% | 0.996 | 754 | 31 | 103 | 237 | 0.803 |
| 1b) 🟦 ResNet-50 3D CNN | ✓ | | | 32.2M | 88.77% | 0.992 | 775 | 25 | 54 | 216 | 0.840 |
| 2b) 🟫 ResNet-50 CNN-LSTM | ✓ | | | 24.6M | 89.74% | 0.996 | 791 | 29 | 38 | 200 | 0.856 |
| 3b) 🟪 ResNet-50 Two-Stream | ✓ | ✓ | | 47.0M | 85.25% | 0.997 | 806 | 49 | 82 | 185 | 0.836 |
| 4b) 🟦 ResNet-50 SlowFast | ✓ | | | 36.7M | 89.01% | 0.987 | 824 | 23 | 83 | 167 | 0.858 |

[a]Frame (appearance) features are *raw frames*; Flow (motion) features are *optical flow computed between adjacent frames*.
[b]Temporal context consists of 16 frames, the last of which is the target frame.
[c]Downsampling to 8 fps causes temporally close events to merge, hence total number of intake gestures in the test set is 991.



(a) Features used (b) Temporal context (c) Model depth

Fig. 5. Comparing model performance in terms of $F_1$ scores. It is apparent that (a) models using frames as features perform better than models using optical flow, (b) models with temporal context tend to perform better than models without, and (c) larger (deeper) models tend to perform better. Models are color-coded according to Table IV.

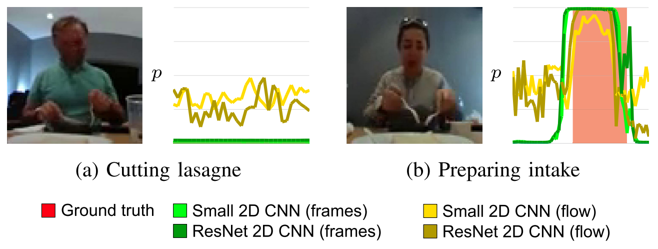


Fig. 6. Example situations showing uncertainty of flow (motion) models compared to frame (appearance) models. Section S1 of the supplementary material provides a multi-frame version of this figure.

the other models with temporal context. We can conclude that for detection of intake gestures, more information is carried by visual appearance than by motion.

*2) Temporal Context, Fig. 5(b):* To assess the usefulness of temporal context, we compare the accuracies of our models with and without temporal context. The straightforward extension of Small 2D CNN to Small 3D CNN adds a 17% relative $F_1$ improvement. Comparing the best models with (ResNet-50 SlowFast) and without temporal context (ResNet-50 2D CNN), we find a relative $F_1$ improvement of 8%. We conclude that temporal context considerably improves model accuracy.

Considering model choice, we observe that the Small 3D CNN is superior to its CNN-LSTM counterpart, however the opposite is true for the ResNet-50 instantiations. This may be due to the fact that for the ResNet instantiations, the CNN-LSTM is pre-trained on ImageNet, while the 3D CNN is not. We conclude that the 3D CNN could be useful for slim models (e.g., for mobile devices), but for larger models, all architectures with temporal context should be considered.

*3) Model Depth, Fig. 5(c):* We also see that the deeper ResNet-50 instantiations achieve higher $F_1$ scores than the small ones for all combinations except the flow-based 2D CNN. Note that the improvement due to model depth is especially noticeable in the $F_1$ score, and less so in UAR.

## D. Why do Frame Features Perform Better?

To help explain why frames perform better as features for this task, we took a closer look at some example model predictions from the validation set. It appears that flows are in general useful as features; however, the data shows that in comparison to frame models, flow models are less certain about their predictions. For example, Fig. 6(a) shows how during periods with no intake gestures, small movements such as using cutlery can cause higher uncertainty in flow models. On the other hand, Fig. 6(b) shows how flow models are also overall less confident when correctly identifying intake gestures. This can also be observed by looking at aggregated predictions for all events in Fig. 7: Models based solely on flows (a) are less certain about predictions, while their predictions also contain more variance than models based on frames (b). Further, this is also reflected in the lower thresholds required to trigger a detection for flow models, as is evident from Table IV. These lower thresholds and uncertainty are linked to the large number of false positives of these models.

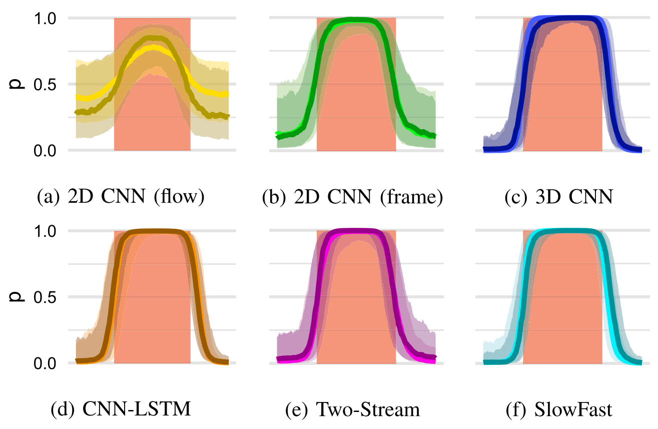


Fig. 7. Aggregating $p_{intake}$ by model for all ground truth events in the validation set. Predictions have been aligned in time and linearly interpolated. We plot the median and $[q_{25}, q_{75}]$ interval for small and ResNet-50 instantiations respectively. Models are color-coded according to Table IV.

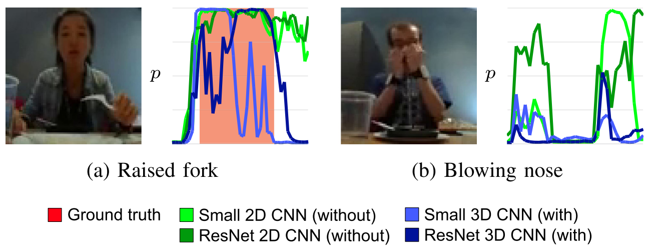


Fig. 8. Example situations where models with temporal context are superior to models without temporal context. Section S2 of the supplementary material provides a multi-frame version of this figure.

## E. Why do Models With Temporal Context Perform Better?

Our results show that while models based on single frames perform reasonably well, there is measurable improvement when adding temporal context. Hence, we also looked at this comparison for example model predictions from the validation set to help make the difference easier interpretable. Indeed, in some cases, it appears intuitive to a human observer how the temporal context is helpful to interpret the target frame. For example, in Fig. 8(a), the participant keeps the fork raised after completing an intake gesture. A frame by itself can seem to be part of an intake gesture, while the participant is actually resting this way or is being interrupted. Without temporal context, the 2D CNN models are unaware of this context, resulting in poor performance. Availability of temporal context also helps models to become more confident in their predictions. Further, errors due to outliers are more easily avoidable with temporal context, such as blowing nose in Fig. 8(b). On the aggregate level, Fig. 7 illustrates how predictions by models with temporal context (c)–(f) have a more snug fit with the ground truth events, and less variance in their predictions.

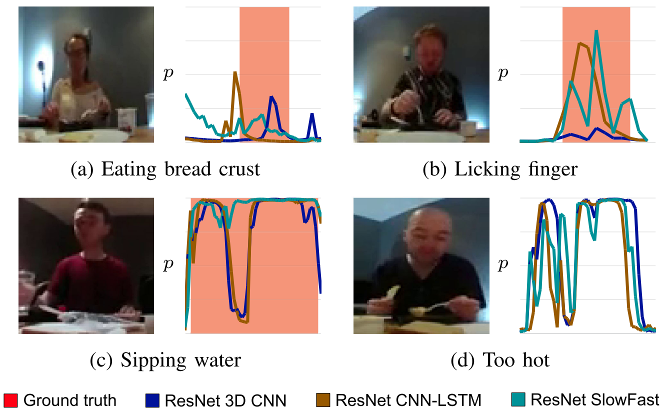


Fig. 9. Example situations where the best models cause false negatives (a–b) and false positives of type one (c) and type two (d). Note that a true positive always preceeds false positives of type one. Section S3 of the supplementary material provides a multi-frame version of this figure.

## F. Where do the Models Struggle?

Examining the results in Table IV, it appears that mainly false negatives and also false positives of type 2 are problematic for our best models. To help understand in what circumstances the models struggle, we compiled examples from the validation set where the best models make mistakes. The examples show that these mistakes tend to happen in cases of "outlier behavior" that differs substantially from the typical behavior in the dataset. False negatives occur mostly for intake gestures that are less noticeable or less common, such as eating bread crust or licking a finger, see Fig. 9(a) and (b). An example for false positives of type 2 is when the participant interrupts an intake gesture as depicted in Fig. 9(d). We see false positives of type 1 as mostly representing a shortcoming of the Stage II approach, i.e., when the duration of an intake gesture exceeds 2 seconds, seen in Fig. 9(c).

## VI. Conclusion

In this paper, we have demonstrated the feasibility of detecting intake gestures from video sourced through a 360-degree camera. Our two-stage approach involves learning frame-level probabilities using deep architectures proposed in the context of video action recognition (Stage I), and a search algorithm to detect individual intake gestures (Stage II). Through evaluation of a variety of models, our results show that appearance features in form of the individual raw frames are well suited for this task. Further, while single frames on their own can lead to useful results with $F_1$ of up to 0.795, the best model considering a temporal context of multiple frames achieves a superior $F_1$ of 0.858. This result is achieved with a state-of-the-art SlowFast network [16] using ResNet-50 as backbone.

Overall, we see several benefits and opportunities that the use of video holds for dietary monitoring. First, the proliferation of 360 degree video reduces the practical challenges of recording images of human eating occasions. This could be used to capture the intake of multiple individuals with a single camera positioned in the center of a table (e.g., families eating from a shared dish [49]). Second, the models could be leveraged to support dietitians in reviewing videos of intake occasions. For instance, instead of watching a twenty minute video, imagery of the actual intake gestures could be automatically extracted for assessment. Third, the models could be used to semi-automate the ground truth annotation process (e.g., for inertial sensors) by pre-annotating the videos. Finally, the models could be used to further the development of fully automated dietary monitoring [7] (e.g., care-taking robots, life-logging, patient monitoring).

As a limitation of our approach, we noted that the distribution of participant behavior has a "fat tail" as it includes many examples of outlier behavior that models misinterpret (e.g., sudden interruption due to a conversation, blowing on food). To deal with such events, future research may employ larger databases of samples to train models. Further, in comparison to approaches based on inertial sensors, our approach has a limitation in that it requires the participant to consume their meal at a table equipped with a camera. Hence, our vision models should be directly benchmarked against models based on inertial sensor data to determine their relative strengths and weaknesses. Going one step further, fusion of both modalities could also be explored. Finally, Stages I and II could be unified into a single end-to-end learning model using CTC loss [50], which may alleviate some of the shortcomings of the current approach. However, it needs to be considered that (i) this is directly only feasible for the CNN-LSTM model without increasing the requirement for GPU memory, and (ii) a larger temporal context and dataset may be required.

## References

[1] C. Weekes *et al.*, "A review of the evidence for the impact of improving nutritional care on nutritional and clinical outcomes and cost," *J. Human Nutrition Dietetics*, vol. 22, pp. 324–335, 2009.

[2] P. V. Rouast, M. T. P. Adam, T. Burrows, R. Chiong, and M. E. Rollo, "Using deep learning and 360 video to detect eating behavior for user assistance systems," in *Proc. Europ. Conf. Inf. Syst.*, 2018, pp. 1–11.

[3] "Noncommunicable diseases progress monitor 2017," World Health Organization, Geneva, Switzerland, Tech. Rep., 2017.

[4] S. W. Lichtman *et al.*, "Discrepancy between self-reported and actual caloric intake and exercise in obese subjects," *New England J. Med.*, vol. 327, no. 27, pp. 1893–1898, 1992.

[5] T. Vu, F. Lin, N. Alshurafa, and W. Xu, "Wearable food intake monitoring technologies: A comprehensive review," *Computers*, vol. 6, no. 1, 2017, Art. no. 4.

[6] K. Kyritsis, C. Diou, and A. Delopoulos, "Modeling wrist micromovements to measure in-meal eating behavior from inertial sensor data," *IEEE J. Biomed. Health Inform.*, to be published, 10.1109/JBHI.2019.2892011.

[7] S. Hantke *et al.*, "I hear you eat and speak: Automatic recognition of eating condition and food type, use-cases, and impact on asr performance," *PloS One*, vol. 11, no. 5, 2016, Art. no. e0154486.

[8] E. Thomaz, I. Essa, and G. D. Abowd, "A practical approach for recognizing eating moments with wrist-mounted inertial sensing," in *Proc. Int. Conf. Ubiquitous Comput.*, 2015, pp. 1029–1040.

[9] M. Mirtchouk, C. Merck, and S. Kleinberg, "Automated estimation of food type and amount consumed from body-worn audio and motion sensors," in *Proc. Int. Conf. Ubiquitous Comput.*, 2016, pp. 451–462.

[10] A. Braeken, P. Porambage, A. Gurtov, and M. Ylianttila, "Secure and efficient reactive video surveillance for patient monitoring," *Sensors*, vol. 16, no. 1, pp. 1–13, 2016.

[11] A. Hall, C. B. Wilson, E. Stanmore, and C. Todd, "Implementing monitoring technologies in care homes for people with dementia: A qualitative exploration using normalization process theory," *Int. J. Nursing Stud.*, vol. 72, pp. 60–70, 2017.

[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[13] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 221–231, Jan. 2013.

[14] J. Donahue *et al.*, "Long-term recurrent convolutional networks for visual recognition and description," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 2625–2634.

[15] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 568–576.

[16] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," 2018, *arXiv:1812.03982*.

[17] G. Ciocca, P. Napoletano, and R. Schettini, "Learning CNN-based features for retrieval of food images," in *Proc. Int. Conf. Image Anal. Process.*, 2017, pp. 426–434.

[18] C. Feichtenhofer, A. Pinz, R. P. Wildes, and A. Zisserman, "What have we learned from deep representations for action recognition?" in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7844–7853.

[19] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 1725–1732.

[20] G. Block, "A review of validations of dietary assessment methods," *Am. J. Epidemiol.*, vol. 115, no. 4, pp. 492–505, 1982.

[21] J. Chen and C.-W. Ngo, "Deep-based ingredient recognition for cooking recipe retrieval," in *Proc. Multimedia Conf.*, 2016, pp. 32–41.

[22] G. Ciocca, P. Napoletano, and R. Schettini, "CNN-based features for retrieval and classification of food images," *Comput. Vis. Image Understanding*, vol. 176, pp. 70–77, 2018.

[23] M. Puri, Z. Zhu, Q. Yu, A. Divakaran, and H. Sawhney, "Recognition and volume estimation of food intake using a mobile device," in *Proc. Workshop Appl. Comput. Vis.*, 2009, pp. 1–8.

[24] W. Zhang, Q. Yu, B. Siddiquie, A. Divakaran, and H. Sawhney, "Snap-n-eat" food recognition and nutrition estimation on a smartphone," *J. Diabetes Sci. Technol.*, vol. 9, no. 3, pp. 525–533, 2015.

[25] Y. Dong, J. Scisco, M. Wilson, E. Muth, and A. Hoover, "Detecting periods of eating during free-living by tracking wrist motion," *IEEE J. Biomed. Health Inform.*, vol. 18, no. 4, pp. 1253–1260, Jul. 2014.

[26] X. Ye, G. Chen, Y. Gao, H. Wang, and Y. Cao, "Assisting food journaling with automatic eating detection," in *Proc. CHI Conf. Extended Abstr. Human Factors Comput. Syst.*, 2016, pp. 3255–3262.

[27] E. Robinson *et al.*, "A systematic review and meta-analysis examining the effect of eating rate on energy intake and hunger," *Amer. J. Clin. Nutrition*, vol. 100, no. 1, pp. 123–151, 2014.

[28] H. Heydarian, M. Adam, T. Burrows, C. Collins, and M. E. Rollo, "Assessing eating behaviour using upper limb mounted motion sensors: A systematic review," *Nutrients*, vol. 11, no. 5, 2019, Art. no. E1168.

[29] O. Amft, M. Stager, P. Lukowicz, and G. Troster, "Analysis of chewing sounds for dietary monitoring," in *Proc. Int. Conf. Ubiquitous Comput.*, 2005, pp. 56–72.

[30] S. Päßler, M. Wolff, and W.-J. Fischer, "Food intake monitoring: An acoustical approach to automated food intake activity detection and classification of consumed food," *Physiol. Meas.*, vol. 33, no. 6, pp. 1073–1093, 2012.

[31] E. S. Sazonov and J. M. Fontana, "A sensor system for automatic detection of food intake through non-invasive monitoring of chewing," *IEEE Sensors J.*, vol. 12, no. 5, pp. 1340–1348, May 2012.

[32] O. Amft, H. Junker, and G. Troster, "Detection of eating and drinking arm gestures using inertial body-worn sensors," in *Proc. IEEE Int. Symp. Wearable Comput.*, 2005, pp. 160–163.

[33] Y. Shen, J. Salley, E. Muth, and A. Hoover, "Assessing the accuracy of a wrist motion tracking method for counting bites across demographic and food variables," *IEEE J. Biomed. Health Inform.*, vol. 21, no. 3, pp. 599–606, May 2017.

[34] S. Zhang, W. Stogin, and N. Alshurafa, "I sense overeating: Motif-based machine learning framework to detect overeating using wrist-worn sensing," *Inf. Fusion*, vol. 41, pp. 37–47, 2018.

[35] J. Gao, A. G. Hauptmann, A. Bharucha, and H. D. Wactlar, "Dining activity analysis using a hidden Markov model," in *Proc. IEEE Int. Conf. Pattern Recognit.*, 2004, pp. 915–918.

[36] K. Okamoto and K. Yanai, “Grillcam: A real-time eating action recognition system,” in *Proc. Int. Conf. Multimedia Model.*, 2016, pp. 331–335.

[37] H. M. Hondori, M. Khademi, and C. V. Lopes, “Monitoring intake gestures using sensor fusion (microsoft kinect and inertial sensors) for smart home tele-rehab setting,” in *Proc. Healthcare Innov. Conf.*, 2012, pp. 36–39.

[38] J. S. Tham, Y. C. Chang, and M. F. A. Fauzi, “Automatic identification of drinking activities at home using depth data from RGB-D camera,” in *Proc. IEEE Int. Conf. Control, Autom. Inf. Sci.*, 2014, pp. 153–158.

[39] A. Klaser, M. Marszałek, and C. Schmid, “A spatio-temporal descriptor based on 3d-gradients,” in *Proc. Brit. Mach. Vis. Conf.*, 2008, pp. 1–10.

[40] H. Wang, A. Kläser, C. Schmid, and L. Cheng-Lin, “Action recognition by dense trajectories,” in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 3169–3176.

[41] J. Carreira and A. Zisserman, “Quo vadis, action recognition? a newmodel and the kinetics dataset,” in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4299–4308.

[42] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, “Learning spatiotemporal features with 3d convolutional networks,” in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 4489–4497.

[43] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[44] C. Feichtenhofer, A. Pinz, and A. Zisserman, “Convolutional two-stream network fusion for video action recognition,” in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1933–1941.

[45] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[46] C. Zach, T. Pock, and H. Bischof, “A duality based approach for realtime tv-l1 optical flow,” *DAGM: Pattern Recognit.*, 2007, pp. 214–223.

[47] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[48] C. Merck, C. Maher, M. Mirtchouk, M. Zheng, Y. Huang, and S. Kleinberg, “Multimodality sensing for eating recognition,” in *Proc. Int. Conf. Pervasive Comput. Technol. Healthcare*, 2016, pp. 130–137.

[49] T. Burrows, C. Collins, M. T. P. Adam, K. Duncanson, and M. Rollo, “Dietary assessment of shared plate eating: A missing link,” *Nutrients*, vol. 11, no. 4, pp. 1–14, 2019.

[50] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks,” in *Proc. 23rd Int. Conf. Mach. Learn.*, 2006, pp. 369–376.