12-10-2021

Eating detection with a head-mounted video camera

Shengjie Bi
Shengjie.Bi@Dartmouth.edu

David Kotz
David.F.Kotz@Dartmouth.EDU

Eating detection with a head-mounted video camera

Shengjie Bi and David Kotz
Dartmouth Computer Science Technical Report TR2021-1002
Dec 10, 2021

Abstract

In this paper, we present a computer-vision based approach to detect eating. Specifically, our goal is to develop a wearable system that is effective and robust enough to automatically detect *when* people eat, and for *how long*. We collected video from a cap-mounted camera on 10 participants for about 55 hours in free-living conditions. We evaluated performance of eating detection with four different Convolutional Neural Network (CNN) models. The best model achieved accuracy 90.9% and F1 score 78.7% for eating detection with a 1-minute resolution. We also discuss the resources needed to deploy a 3D CNN model in wearable or mobile platforms, in terms of computation, memory, and power. We believe this paper is the first work to experiment with video-based (rather than image-based) eating detection in free-living scenarios.

1 Introduction

Convolutional Neural Networks (CNN) have been established as a powerful method for image recognition and action recognition in videos [1–5]. Encouraged by these results, we applied CNN to eating detection using vision-based approaches. We used a miniature head-mounted camera for data collection and then (offline) trained CNN models for eating detection using images and videos, respectively. The camera is fixed under the brim of a cap, pointing to the mouth of participants (as shown in Figure 1) and continuously recording video (but not audio) throughout their normal daily activity. Bi et al. used such a camera for collecting ground truth in their field study [6]; we now use that video itself as a more accurate (and more comfortable) way to detect eating.

In recent years, similar systems have been widely used to collect ground truth for field studies with Automatic Dietary Monitoring (ADM) systems [7–9]. In the future, researchers may be able to run the ground-truth videos they collected using our proposed approach and compare the performance of their approach with ours, if they use similar methods for ground-truth collection. This opportunity could address one of the major challenges in the field of ADM – the lack of comparison between different approaches [10]. Furthermore, our approach could assist in video annotation and thus reduce the video-annotation burden in the field of ADM.

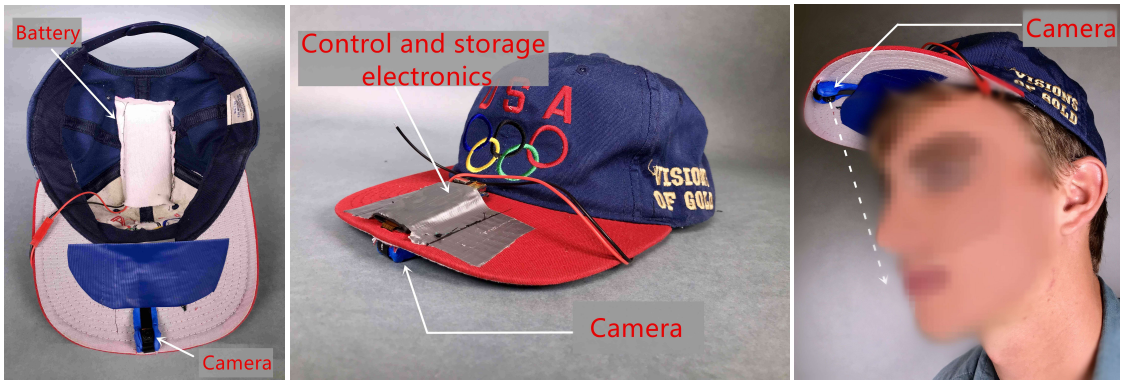

Figure 1: Head-mounted camera; Figure from [6]

Figure 2: Cap after adjusting the battery location

This paper makes the following **contributions**:

- We developed the first video-based (rather than image-based [11, 12]) approach for eating detection in free-living conditions and demonstrated its success in a field deployment involving 10 participants.
- We demonstrated the feasibility of using CNN models to detect eating from raw video frames of a face viewed from an oblique angle.
- We showed that temporal context is crucial and considerably improved the performance for eating detection in free-living conditions, when using CNN models.

2 System design

We refined the miniature cap-mounted camera (shown in Figure 1) and then used it for collecting more data (videos). The resolution and frame rate of the video recorded by this camera is 360p (640×360 pixels) and 30 frames per second (FPS), which we found to be sufficient for our data analysis.

To enable longer data-collection sessions (as described in Section 3), we improved device comfort by adjusting the size and location of battery and the pocket holding the battery (shown in Figure 2), and pilot-tested various arrangements with prospective participants to find a design that fit comfortably for most, if not all, potential participants. We developed two identical sets of devices so we could collect data with one while we were sanitizing the other, or collect data in parallel on two participants.

Figure 3: video frame examples recorded during a eating period

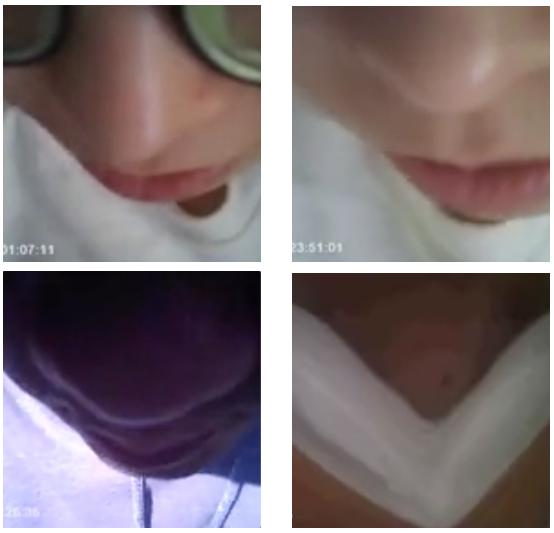

Figure 4: video frame examples recorded during non-eating periods

3 Data collection

Under protocols approved by our Institutional Review Board (IRB), we collected data using the system described in Section 2 in free-living scenarios.

We collected data from 10 participants (4 female, 6 male) for total 55 hours. During these periods of field data acquisition, participants ate various types of food including rice, bread, noodles, meat, vegetables, fruit, eggs, nuts, chips, soup, and ice cream. Participants recorded data in diverse environments including houses, cars, parking lots, restaurants, kitchens, woods, and streets.

After a preliminary review of data, we determined that we could not use data for about 4 hours from 2 participants, for the following reasons. For one participant, wheat flour used by the participant during cooking accidentally pasted on the camera and blurred two hours of video recorded. For another participant, the participant pressed the brim of the cap to a low position for two hours, so the camera did not capture the mouth, cheek or chin – only part of the nose. Our analysis excludes these 4 hours. We used the remaining 51 hours of recorded video from 10 participants for further analysis. Figure 3 and Figure 4 show examples of video frames recorded during eating and non-eating periods, respectively.

We followed a data collection protocol much like that in Bi et al., except the participants were only asked to wear the cap; they did not wear the Auracle device [6]. Before the study, we told each participant they were free to remove cap when they need privacy. Because most people spend a relatively small fraction of their day eating, we strongly encouraged all the participants to eat more often than usual during our study. Additionally, to collect more eating data during our studies, we increased the session duration to be 5 hours (or longer), which included two meals for each participant.

For video annotation, we used the same commercial service to annotate the videos as the one used by Bi et al. [13]. The annotation process consists of three steps: execution, audit, and quality inspection. More details about each step can be found in the work of Bi et al. [14].

4 Data analysis

We next describe our evaluation metrics, and the stages of our data-processing pipeline: preprocessing, classification, and aggregation.

4.1 Evaluation Metrics

To better compare with Bi et al. [6], we used the same evaluation metrics. However, to reduce the computation required for training a CNN model, we performed a global split of our video dataset rather than using a Leave-One-Person-Out (LOPO) cross-validation. Note that global split is a common and widely used approach in evaluation of CNN, considering the high computation cost of training deep-learning models [1–5,15]. We split our video dataset into three subsets: training, validation, and test. We used the training subset for training the CNN models mentioned in Section 4.3, the validation subset for tuning the parameters of these models, and the test subset for evaluation. The ratio of the total duration of videos is 70:15:15.

4.2 Data preprocessing

To reduce computational burden, we downsample the video from 30 FPS to 5 FPS, and resize from dimensions 640×360 pixels to 256×144 pixels. Because CNN models usually take inputs in square shape, and to further reduce the memory burden, we cropped the downsampled videos to extract the central 144×144 pixels.

For the cropped videos, we used the TensorFlow library to extract all raw video frames (appearance feature) and optical flow (motion feature) and stored them in *tensorflow record* format for faster model training speed [16]. We used three red, green, blue (RGB) channels for raw video frames. We used Dual TV-L1 optical flow because it can be efficiently implemented on a modern graphics processing unit (GPU) [17]. The optical flow is calculated based on the target frame and the frame directly preceding it, and produces two channels corresponding to the horizontal and vertical components.

4.3 Classification

We developed 2-class CNN models to classify *eating* and *non-eating* using the tensorflow records we extracted above. The CNN models output a probability of *eating* for each frame (every 0.2 seconds). We ran experiments with three types of CNN architectures: 2D CNN, 3D CNN, and SlowFast (see Table 1 for model specification). With an eye to deploying the models on wearable platforms, we deliberately selected small CNN models with relatively few parameters. We adopted the five-layer CNN architecture popularised by AlexNet for 2D CNN and 3D CNN models, which includes 4 conventional layers (each with a pooling layer after) and 1 fully connected (dense) layer [18]. For the SlowFast model, there is one more fusion layer between the last pooling layer and the fully connected layer to combine the slow and fast pathways (see Table 1). We adopted and adjusted the model implementation and training policy based on the work of Rouast et al. [19].

4.3.1 2D CNN

We explored with two types of input features: raw video frames or precalculated optical flows. When using raw video frames as input features, the CNN model makes predictions based on the appearance information extracted from only one image segmented from videos (i.e., one video frame); the CNN model produces one inference for each frame, independently of its classification of other frames. Because the 2D CNN model is simpler than the other two models – it uses only one frame or optical flow as the input – we anticipate it will use less memory and computation power when deploying on wearables. Additionally, 2D CNN functions as a baseline for our study, indicating what is possible with only appearance information or motion information. We used max pooling for all the pooling layers.

Table 1: CNN model specifications. For 2D CNN, colors red and cyan show the difference between using frame and flow. For SlowFast, colors blue and *magenta* show the difference between the slow and fast pathways.

Layer	**2D CNN** (with frame or flow)			**3D CNN**			**SlowFast** (slow + *fast*)		
	dimension	kernel size	stride	dimension	kernel size	stride	dimension	kernel size	stride
data	$128^2 \times 3\|2$			$16 \times 128^2 \times 3$			$4 \times 128^2 \times 3$ $16 \times 128^2 \times 3$		
conv1	$128^2 \times 32$	3^2	1^2	$16 \times 128^2 \times 32$	3×3^2	1×1^2	$4 \times 128^2 \times 32$ $16 \times 128^2 \times 8$	$1\|3 \times 3^2$	1×1^2
pool1	$64^2 \times 32$	2^2	2^2	$8 \times 64^2 \times 32$	2×2^2	2×2^2	$4 \times 64^2 \times 32$ $16 \times 64^2 \times 8$	1×2^2	1×2^2
conv2	$64^2 \times 32$	3^2	1^2	$8 \times 64^2 \times 32$	3×3^2	1×1^2	$4 \times 64^2 \times 32$ $16 \times 64^2 \times 8$	$1\|3 \times 3^2$	1×1^2
pool2	$32^2 \times 32$	2^2	2^2	$4 \times 32^2 \times 32$	2×2^2	2×2^2	$4 \times 32^2 \times 32$ $16 \times 32^2 \times 8$	1×2^2	1×2^2
conv3	$32^2 \times 64$	3^2	1^2	$4 \times 32^2 \times 32$	3×3^2	1×1^2	$4 \times 32^2 \times 64$ $16 \times 32^2 \times 16$	$1\|3 \times 3^2$	1×1^2
pool3	$16^2 \times 64$	2^2	2^2	$1 \times 16^2 \times 64$	2×2^2	2×2^2	$4 \times 16^2 \times 64$ $16 \times 16^2 \times 16$	1×2^2	1×2^2
conv4	$16^2 \times 64$	3^2	1^2	$2 \times 16^2 \times 64$	3×3^2	1×1^2	$4 \times 16^2 \times 64$ $16 \times 16^2 \times 16$	$1\|3 \times 3^2$	1×1^2
pool4	$8^2 \times 64$	2^2	2^2	$1 \times 8^2 \times 64$	2×2^2	2×2^2	$4 \times 8^2 \times 64$ $16 \times 8^2 \times 16$	1×2^2	1×2^2
fusion							$8^2 \times 64$		
flatten	4096			4096			4096		
dense	1024			1024			1024		
dense	2			2			2		

4.3.2 3D CNN

A 3D CNN has the ability to learn spatio-temporal features as it extends the 2D CNN introduced in the previous section by using 3D instead of 2D convolutions [1]. The third dimension corresponds to the temporal context. The input of 3D CNN consists the target frame and the 15 frames preceding it (3 seconds at 5 FPS), which are a sequence of 16 frames in total. In other words, the 3D CNN considers a consecutive stack of 16 video frames. The output of the CNN model is the prediction for the last frame of the sequence (the target frame). To take maximum advantage of the available training data, we generated input using a window shifting by one frame. We used temporal convolution kernels of size 3 as suggested by Tran et al. [4]. We used max pooling for the temporal dimension in all the pooling layers.

4.3.3 SlowFast

Similar to the 3D CNN, the SlowFast model also considers a temporal context of the previous frames preceding the target frame, but the SlowFast model processes the temporal context at two different temporal resolutions. As recommended by Rouast et al. [19], we chose the factors $\alpha = 4$, temporal kernel size 3 for the fast pathway, and $\beta = 0.25$, temporal kernel size 1 for the slow pathway. We adopted the method for developing the fusion layer from the work by Feichtenhofer et al. [15].

4.3.4 Model training policy

We used the Adam optimizer to train each model on the training set and chose batch size 64 based on the memory size of the cluster we used. Training ran for 40 epochs with a learning rate starting at 2×10^{-4} and exponentially decaying at a rate of 0.9 per epoch.

We used cross entropy for loss calculation for all our models. Due to the nature of our data, the classes are imbalanced with more *non-eating* instances than *eating* instances. When training our models, we corrected this imbalance by scaling the weight of loss for each class using the reciprocal of number of instances in each class. For example, in a batch of training samples (size 64) with 54 *non-eating* instances and 10 *eating* instances, the ratio of weight of loss between *non-eating* class and *eating* class is $10 : 54$.

To avoid over fitting, we used L2 loss with a lambda of 1×10^{-4} for regularization and applied dropout in all models on convolutional and dense layers with rate 0.5. Additionally, we used early stopping if we observed the model yields increasing validation errors at the end of the training stage. We also used data augmentation by applying random transformations to the input: cropping to size 128×128, horizontal flipping, small rotations, brightness and contrast changes. Among these transformations, brightness and contrast changes can help a model better deal with eating detection in various light conditions. All models were learned end to end.

4.4 Aggregation

We applied the same aggregation approach as the stage-A aggregation used by Bi et al. to the classifier outputs and to ground-truth labels [6]. We also applied the same rule for aggregation: if more than 10% of the windows in a minute were labeled eating, we labeled that minute as eating. Before aggregation, the models output every 0.2 seconds (one inference per frame). After aggregation, the models output one inference per minute.

5 Performance evaluation

Table 2 summarizes the resulting performance metrics for eating detection with a 1-minute resolution using the four models. We achieved the best result using SlowFast model, with an F1 score of 78.7% and accuracy

Table 2: Performance metrics for eating detection with CNN models.

Model	#Parameters	Accuracy	Precision	Recall	F1 Score
2D CNN (with frame)	4.26M	71.0%	38.3%	49.8%	43.3%
2D CNN (with flow)	4.26M	78.3%	46.9%	67.8%	55.4%
3D CNN	4.39M	86.4%	72.4%	75.3%	73.8%
SlowFast	4.49M	90.9%	75.5%	82.2%	78.7%

of 90.9%.

To assess the usefulness of temporal context, we compare the accuracy of our models with and without temporal context. Based on Table 2, the 3D CNN model (F1 score 73.8%) outperforms 2D CNN (with frame; F1 score 43.3%) and 2D CNN (with flow; F1 score 55.4%). The SlowFast model also outperforms 2D CNN (with frame) and 2D CNN (with flow) by more than 23% F1 score. We thus conclude that (1) temporal context is crucial for eating detection in the field and considerably improves model performance; (2) using only spatial information (either frame (appearance) or flow (motion) feature) from one single video frame may be not sufficient for achieving good eating-detection performance.

Additionally, we noticed that precision is the worst score across all the metrics for all the four models we experimented. The low precision indicates there were many false positives (the model indicated eating and ground truth indicated non-eating). To identify the reasons, we checked the video frames during the periods that false positives occurred; scenarios include talking, drinking, blowing nose, putting on face masks, mouth rinsing, wiping mouth with napkin, unconscious mouth or tongue movement, and continuously touching face or mouth. We anticipate more training data and deeper CNN networks would help to reduce false positives.

6 Computation, memory, and power

Based on the performance evaluation in Section 5, we found both the 3D CNN and SlowFast models achieved better performance than the 2D CNN models for eating detection. However, the SlowFast model is a fusion of two 3D CNN models so we assume it requires more computational resources than a single 3D CNN model. In this section, we thus focus on whether it is feasible to deploy the 3D CNN model on a mobile or wearable platform, when considering computation, memory, and power constraints.

The computational resources needed for a deep-learning model is often measured in *gigaflops:* 1×10^9 floating point operations per second (GFLOPs). Niu et al. measured a 3D CNN model having 8 convolutional layers and found the overall model requires from 10.8 to 15.2 GFLOPs, after compression with pruning algorithms [20]. We used a 3D CNN model with 4 convolutional layers; thus our model should require less than 10.8 GFLOPs after pruning.

We then investigated GPUs used in modern mobile or wearable platforms. The Google Pixel 3 smartphone has a Qualcomm Adreno 630 GPU that can support 727 GFLOPs [21]. Many modern smartwatches and similar wearable platforms have GPUs as well. For instance, the Huawei Watch GT 2 includes a Qualcomm Adreno 304 GPU that supports 19.2 GFLOPs [21]. Both these platforms have enough computing resources to run our 3D CNN model for inference; we thus conclude that modern mobile or wearable platforms can support the models described in Section 4.3.

The memory needed for running the 3D CNN models include at least two parts: storing the raw video frame sequence, and storing the model parameters. The pixel values of RGB images are integers and the model parameters are floating-point numbers, which (in our implementation) are 4 bytes each. Using the data dimensions from Table 1, the memory needed for storing the raw video frame sequence is $16 \times 128^2 \times 3 \times 4 =$

3.15 MB. Using the parameters from Table 2, the memory needed for storing the parameters of the 3D CNN model is $4.39 \times 4 = 17.56$ MB. Hence the memory needed for running the 3D CNN model is at about $3.15 + 17.56 = 20.71$ MB, and should fit easily in a mobile platform with 32 MB of main memory. Such platforms are readily available and suitable for small wearable devices today; the Apple Watch series 6 has 1000 MB [22].

The power consumption of the system consists of at least two parts: the camera (to capture images or videos) and the processor (to run the CNN model). We investigated ultra-low power CMOS cameras in the literature; a camera with parameters similar to ours (96×96 pixels, 20 FPS) consumes less than $20\,\mu\mathrm{W}$ [23]. We conclude that the power consumption for capturing images or videos can be ignored, if using an ultra-low power CMOS camera that is specifically designed as needed.

We found little information, however, regarding the power consumption of GPUs used in mobile or wearable platforms (e.g., Qualcomm Adreno 304). We were only able to find that mobile GPUs are typically designed for a power ceiling under $1\,\mathrm{W}$ [24]. Given this assumption, the upper limit of power consumption for continuous running the 3D CNN model for a waking day (16 hours) is $16\,\mathrm{W\,h}$, which is $4234\,\mathrm{mA\,h}$ when the voltage is $3.7\,\mathrm{V}$. To address this need, we could use two *18650 lithium-ion batteries* (e.g., Samsung 35E 18650 battery) as the power supply for the system, which have enough capacity ($7000\,\mathrm{mA\,h}$ in total) and are cheap ($5–10 each), small (18.75 mm $\times$ 65.25 mm), and rechargeable [25]. Note that this is estimate is only a rough upper bound on GPU power consumption. A GPU that can run our model does not need to be powered at $1\,\mathrm{W}$ and the GPU does not necessarily need to continuously run for 16 hours. For instance, during some periods users may be sitting quietly at a desk while studying, so there is no movement captured by the camera. These periods can be easily filtered out as *non-eating* and we can set the GPU to idle mode to save power, much like the lower-power 'sleep' state used by Bi et al [6].

Because modern mobile phones often have a powerful GPU, it maybe beneficial to transmit the video frames from the cap to the mobile phone for running 3D CNN models – assuming current Bluetooth technology can support the necessary data-transmission rate. The video we used is 5 FPS so our system needs to transmit $5 \times 128^2 \times 3 \times 4 = 0.98$ MB per second, which is about 8 megabit per second (Mbps). The new Bluetooth 5.0 technology can support a data transfer rate as high as 50 Mbps, so it may indeed feasible to take this approach [26]. Further investigation would be necessary to consider the power tradeoff between on-board GPU processing vs. Bluetooth transfer to the phone for processing. Privacy is another potential issue, as the export of raw video from the cap to the phone poses a potential risk for that video being obtained by network eavesdroppers or malware based in the phone.

7 Future work

Here we discuss ideas worth exploring.

Detection of drinking and other health-related behaviours: CNN models have been widely used for the recognition of various human actions in videos [3–5]. With enough training data and proper model tuning, our method may generalize to the detection of other health-related behaviours (such as drinking, smoking, coughing, sniffling, laughing, breathing, speaking, and face touching). However, most of these behaviors are usually short and infrequent during normal daily life, so large-scale field studies (and substantial video annotation effort) may be necessary to collect enough training data.

Images and videos with different key parameters: In this project, we only experimented with RGB videos frames that have relatively low resolution (144×144 pixels) and low frame rate (5 FPS) due to limited computation resources. It would be interesting to explore parameters (i.e., frame rate, frame resolution, color depth) that affect cost (e.g., power consumption) and performance (e.g., F1 score) of the approaches we used, and characterize the trade-offs between cost and performance as these parameters change.

Fusion of visual and privacy-sensitive audio signals: Researchers have developed acoustic-based ADM systems for eating detection and showed that audio signals (e.g., chewing sound and swallowing sound) are useful for eating detection [6, 7, 27]. Our system is located close to the face and can be easily modified to capture both video and audio signals. In our experiment, we chose not to collect audio due to privacy concerns. A module that could process audio on-board could address this issue [28]. Thus, it is worth investigating the fusion of visual and privacy-sensitive audio signals, which may yield better performance in eating detection.

Deeper CNN networks: If experimenting with deeper CNN networks, the performance of eating detection may further improve. There exist implementations of many pre-trained deeper networks, such as ResNet and GoogleNet, that could be used to initialize a model that could then be fine-tuned for eating detection [29, 30]. Specifically, it is worth exploring these deeper networks as a backbone for the 3D CNN and SlowFast models, to see how much improvement the deeper networks can achieve.

Different types of cameras: In this study, we developed an eating-detection approach using a traditional digital camera and Computer Vision (CV) techniques. Other types of cameras (e.g., thermal cameras and event cameras) could also be useful sensors for eating detection. Thermal cameras could take advantage of the temperature information from food and use it as a cue for eating detection. Event cameras contain pixels that independently respond to changes in brightness as they occur [31]. Compared with traditional cameras, event cameras have several benefits including extremely low latency, asynchronous data acquisition, high dynamic range, and very low power consumption [32], which make them interesting sensors to explore for eating detection in the future.

Explainability of CNN model: The development of methods for visualizing, explaining and interpreting deep learning models has recently attracted increasing attention [33]. One of the most popular methods is the use of heatmaps to visualize the importance of each pixel for the prediction. Similar explanation methods could help us to understand the reason our models arrived at a specific decision, so we could further improve our eating-detection approaches accordingly.

8 Related work

Researchers have explored various types of deep-learning architectures for action recognition in videos; four architectures are widely used, as shown in Figure 5. Here we give one example for each of them. Tran et al. proposed 3D CNN to address the problem of learning spatiotemporal features on large-scale video dataset [4]. They evaluated their approach on the UCF-101 dataset, which consists of 13,320 videos of 101 human action categories, and achieved 85.2% accuracy when taking RGB frames as inputs. Donahue et al. developed a CNN-long short-term memory (LSTM) model, which uses the sequence of spatial features learned by a CNN from individual video frames as input into a LSTM Recurrent Neural Network (RNN) [34]. The CNN-LSTM model has the advantage of being more flexible with regards of the number of input frames, but appears to require more training data in comparison to other approaches [19]. Simonyan et al. proposed a two-stream CNN architecture that incorporates spatial and temporal networks [2]. This architecture models short temporal snapshots of videos by averaging the predictions from a single RGB frame and a stack of 10 externally computed optical-flow frames [5]. They demonstrated that this method can achieve high performance on existing benchmarks, while being efficient to train and test [5]. Feichtenhofer et al. proposed the 'SlowFast' architecture, which involves (1) a *slow* pathway, operating at low frame rate, to capture spatial semantics, and (2) a *fast* pathway, operating at fast frame rate, to capture motion at fine temporal resolution [35]. They reported state-of-the-art accuracy on several major video-recognition benchmarks.

In recent years, researchers have explored developing ADM systems based on videos and CV techniques. Rouast et al. explored video-based intake gesture detection, which is the closest related work we found in

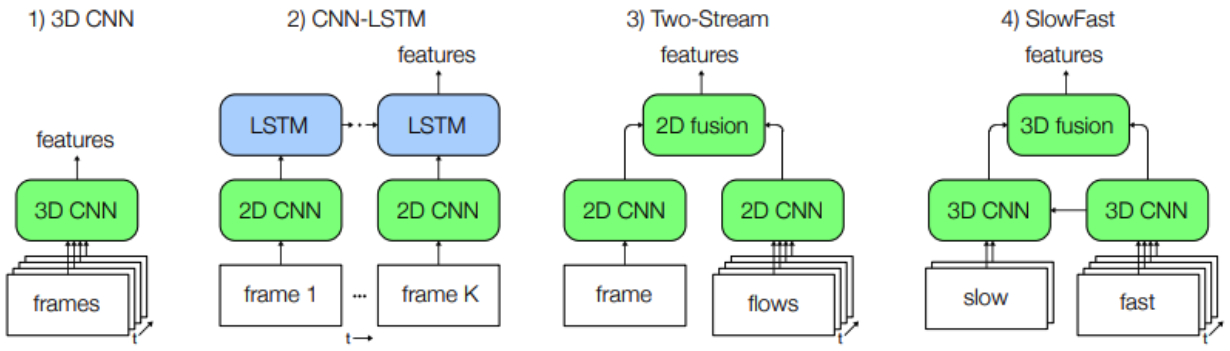

Figure 5: Four architectures for action recognition in videos; figure from [19].

the literature [19, 36]. They placed a 360-degree camera on a dining table and simultaneously recorded four participants seated around the table during a meal. In total, they collect and label video data of 102 participants in a laboratory setting. They experimented with four different architectures and achieved the best F1 score (0.858) with the SlowFast model. Their work differs from ours in three ways. First, they focused only on the detection of intake gestures rather than entire eating activities, which are a combination of one or more intake gestures, chewing, and swallowing. Second, we used a vision-based approach to detect eating in free-living scenarios from raw video frames of a face viewed by a wearable camera from an oblique angle, while they evaluated their method using third-person videos collected from a fixed camera in a laboratory setting. Third, their work focused only on evaluating performance of intake detection with different CNN models. We also validated the feasibility of deploying 3D CNN models in wearable or mobile platforms with computation, memory, and power constraints.

In other work, Qiu et al. proposed an approach to count the number of bites and recognize consumed food items in egocentric videos [37]. In their experiments, they achieved 74.15% top-1 accuracy (classifying between 0–4 bites in 20-second clips) for counting bites and 40.5% F1 score for recognizing 66 types of different consumed food items. Our work differs in three main ways. First, our goal is different from theirs: we focused on the detection of eating episodes while their goal is bite counting and food recognition during eating episodes. Second, their data collection is in laboratory conditions while our study is in free-living scenarios. Third, they focused only on a performance evaluation of deep neural networks. We did both a performance evaluation and a computation, memory and power evaluation of CNN models.

9 Conclusion

In this paper, we developed a computer-vision based approach for eating detection. Indeed, we believe this paper represents **the first work to experiment with video-based (rather than image-based) eating detection in free-living scenarios**. Using a miniature head-mounted camera, we conducted a field study and collected data with 10 participants for about 55 hours. We designed and evaluated performance of eating detection using four different CNN models. The best model achieved an accuracy 90.9% and an F1 score 78.7% for eating detection with a 1-minute resolution. Finally, we discussed the feasibility of deploying the 3D CNN model in wearable or mobile platforms with computation, memory, and power constraints.

Acknowledgements

I am grateful for the help from John Hudson and Arnold Song in using the Discovery cluster and acknowledge Philipp Rouast and Marc Adam for making their code and dataset publicly available to research community [19, 36].

This research results from a research program at the Center for Technology and Behavioral Health (CTBH) at Dartmouth College, supported by the National Science Foundation under award numbers CNS-1565269, CNS-1835983, CNS-1565268, and CNS-1835974. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors.

References

[1] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale Video Classification with Convolutional Neural Networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, p. 1725–1732, DOI 10.1109/CVPR.2014.223.

[2] K. Simonyan and A. Zisserman, "Two-Stream Convolutional Networks for Action Recognition in Videos," in *Advances in neural information processing systems*, 2014, pp. 568–576, DOI 10.5555/2968826.2968890.

[3] D. Tran, H. Wang, L. Torresani, J. Ray, Y. Lecun, and M. Paluri, "A Closer Look at Spatiotemporal Convolutions for Action Recognition," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 6450–6459, 2018, DOI 10.1109/CVPR.2018.00675.

[4] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2015 Inter, pp. 4489–4497, 2015, DOI 10.1109/ICCV.2015.510.

[5] J. Carreira and A. Zisserman, "Quo Vadis, action recognition? A new model and the kinetics dataset," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 4724–4733, 2017, DOI 10.1109/CVPR.2017.502.

[6] S. Bi, T. Wang, N. Tobias, J. Nordrum, S. Wang, G. Halvorsen, S. Sen, R. Peterson, K. Odame, K. Caine, R. Halter, J. Sorber, and D. Kotz, "Auracle: Detecting Eating Episodes with an Ear-Mounted Sensor," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT) (Ubicomp)*, vol. 2, no. 3, Sep. 2018, DOI 10.1145/3264902.

[7] A. Bedri, R. Li, M. Haynes, R. P. Kosaraju, I. Grover, T. Prioleau, M. Y. Beh, M. Goel, T. Starner, and G. Abowd, "EarBit: Using Wearable Sensors to Detect Eating Episodes in Unconstrained Environments," *Proc. ACM Interactive, Mobile and Wearable Ubiquitous Technology*, vol. 1, no. 3, Sep. 2017, DOI 10.1145/3130902.

[8] S. Zhang, Y. Zhao, D. T. Nguyen, R. Xu, S. Sen, J. Hester, and N. Alshurafa, "NeckSense: A Multi-Sensor Necklace for Detecting Eating Activities in Free-Living Conditions," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT) (Ubicomp)*, vol. 4, no. 2, Jun. 2020, DOI 10.1145/3397313.

[9] A. Bedri, D. Li, R. Khurana, K. Bhuwalka, and M. Goel, "FitByte: Automatic Diet Monitoring in Unconstrained Situations Using Multimodal Sensing on Eyeglasses CCS Concepts,"

in *CHI Conference on Human Factors in Computing Systems*, vol. 20, 2020, pp. 1–12, DOI 10.1145/3313831.3376869.

[10] B. M. Bell, R. Alam, N. Alshurafa, J. Lach, and D. Spruijt-metz, "Automatic , wearable-based , in-field eating detection approaches for public health research : a scoping review," *npj Digital Medicine*, 2020, DOI 10.1038/s41746-020-0246-2.

[11] D. Castro, E. Thomaz, I. Essa, S. Hickson, G. Abowd, V. Bettadapura, and H. Christensen, "Predicting daily activities from egocentric images using deep learning," *Proceedings of the 2015 ACM International Symposium on Wearable Computers (ISWC)*, pp. 75–82, Sep. 2015, DOI 10.1145/2802083.2808398.

[12] D. Hossain, M. H. Imtiaz, T. Ghosh, V. Bhaskar, and E. Sazonov, "Real-Time Food Intake Monitoring Using Wearable Egocentric Camera," *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*, pp. 4191–4195, Jul. 2020, DOI 10.1109/EMBC44109.2020.9175497.

[13] (2021, April) Basicfinder annotation service. Online at https://www.basicfinder.com/en/.

[14] S. Bi, Y. Lu, N. Tobias, E. Ryan, T. Masterson, S. Sen, R. Halter, J. Sorber, D. Gilbert-Diamond, and D. Kotz, "Measuring children's eating behavior with a wearable device," in *2020 IEEE International Conference on Healthcare Informatics (ICHI)*, 2020, pp. 1–11, DOI 10.1109/ICHI48887.2020.9374304.

[15] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional Two-Stream Network Fusion for Video Action Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, DOI 10.1109/CVPR.2016.213.

[16] TensorFlow. (2021, April) Tensorflow website. Online at https://www.tensorflow.org/.

[17] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime TV-L1 optical flow," in *DAGM conference on Pattern recognition*, 2007, pp. 214–223, Online at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.709.4597&rep=rep1&type=pdf.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, Jun. 2017, DOI 10.1145/3065386.

[19] P. V. Rouast and M. Adam, "Learning deep representations for video-based intake gesture detection," *IEEE Journal of Biomedical and Health Informatics*, vol. PP, no. 8, pp. 1–1, 2019, DOI 10.1109/jbhi.2019.2942845.

[20] W. Niu, M. Sun, Z. Li, J.-A. Chen, J. Guan, X. Shen, Y. Wang, S. Liu, X. Lin, and B. Ren, "RT3D: Achieving Real-Time Execution of 3D Convolutional Neural Networks on Mobile Devices," *arXiv*, Jul. 2020, Online at http://arxiv.org/abs/2007.09835.

[21] Wikipedia. (2021, April) Wikipedia for adreno GPU. Online at https://en.wikipedia.org/wiki/Adreno.

[22] ——. (2021, April) Wikipedia for Apple Watch series 6. Online at https://en.wikipedia.org/wiki/Apple_Watch_Series_6.

[23] I. Cevik, X. Huang, H. Yu, M. Yan, and S. Ay, "An Ultra-Low Power CMOS Image Sensor with On-Chip Energy Harvesting and Power Management Capability," *Sensors*, vol. 15, no. 3, pp. 5531–5554, Mar. 2015, DOI 10.3390/s150305531.

[24] K. T. Cheng and Y. C. Wang, "Using mobile GPU for general-purpose computing a case study of face recognition on smartphones," in *Proceedings of 2011 International Symposium on VLSI Design, Automation and Test, VLSI-DAT 2011*, 2011, pp. 54–57, DOI 10.1109/VDAT.2011.5783575.

[25] (2021, April) 18650 battery store. Online at https://www.18650batterystore.com/collections/samsung-18650-batteries.

[26] (2021, April) Bluetooth: everything you need to know about the popular wireless standard. Online at https://www.ionos.com/digitalguide/server/know-how/bluetooth/.

[27] R. Zhang and O. Amft, "Monitoring Chewing and Eating in Free-Living Using Smart Eyeglasses," *IEEE Journal of Biomedical and Health Informatics*, vol. 22, no. 1, pp. 23–32, Jan. 2018, DOI 10.1109/jbhi.2017.2698523.

[28] R. Wang, F. Chen, Z. Chen, T. Li, G. Harari, S. Tignor, X. Zhou, D. Ben-Zeev, and A. T. Campbell, "Studentlife: Assessing mental health, academic performance and behavioral trends of college students using smartphones," in *UbiComp 2014 - Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. New York, NY, USA: Association for Computing Machinery, Inc, Sep. 2014, pp. 3–14, DOI 10.1145/2632048.2632054.

[29] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June-2015. IEEE Computer Society, Oct. 2015, pp. 1–9, DOI 10.1109/CVPR.2015.7298594.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-December. IEEE Computer Society, Dec. 2016, pp. 770–778, DOI 10.1109/CVPR.2016.90.

[31] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 $\times$ 128 120 dB 15 μs latency asynchronous temporal contrast vision sensor," *IEEE Journal of Solid State Circuits*, vol. 43, no. 2, pp. 566–576, 2008, DOI 10.1109/JSSC.2007.914337.

[32] A. Z. Zhu, D. Thakur, T. Özaslan, B. Pfrommer, V. Kumar, and K. Daniilidis, "The Multivehicle Stereo Event Camera Dataset: An Event Camera Dataset for 3D Perception," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2032–2039, Jul. 2018, DOI 10.1109/LRA.2018.2800793.

[33] W. Samek, T. Wiegand, and K.-R. Müller, "Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models," *ITU Journal: ICT Discoveries*, vol. 1, no. No.1, pp. 39–48, 2018, Online at https://www.itu.int/en/journal/001/Pages/05.aspx.

[34] J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell, "Long-Term Recurrent Convolutional Networks for Visual Recognition and Description," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 677–691, 2017, DOI 10.1109/TPAMI.2016.2599174.

[35] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2019-Octob, pp. 6201–6210, 2019, DOI 10.1109/ICCV.2019.00630.

[36] P. V. Rouast, H. Heydarian, M. T. Adam, and M. E. Rollo, "OReBA: A dataset for objectively recognizing eating behavior and associated intake," *IEEE Access*, vol. 8, pp. 181 955–181 963, 2020, DOI 10.1109/ACCESS.2020.3026965.

[37] J. Qiu, F. P. W. Lo, S. Jiang, C. Tsai, Y. Sun, and B. Lo, "Counting Bites and Recognizing Consumed Food from Videos for Passive Dietary Monitoring," *IEEE Journal of Biomedical and Health Informatics*, pp. 1–1, Sep. 2020, DOI 10.1109/jbhi.2020.3022815.