Perceptually-enabled Task Guidance and Knowledge Transfer

David Weinflash
University of California Santa Barbara
dweinflash@cs.ucsb.edu

Misha Sra
University of California Santa Barbara
sra@cs.ucsb.edu

Tobias Höllerer
University of California Santa Barbara
holl@cs.ucsb.edu

ABSTRACT

The most effective virtual assistants help users perform tasks within and beyond their skill set. Any domain specific knowledge held by the assistant is transferred to the user in the form of intuitive, timely advice. Ultimately, feedback from the assistant is both helpful and nonintrusive. In the following research project, we investigate the effectiveness of baseline action recognition models in enabling non-intrusive task guidance. Through offline and real-time video analysis, we demonstrate knowledge transfer between assistant and user by way of our augmented reality application, *The AR Cooking Helper*. Using contemporary action recognition models trained on the EPIC-KITCHENS dataset, we walk a user through a recipe, issuing real-time guidance when necessary. Additionally, to best inform the virtual assistant, we explore techniques in image tracking and gaze interaction to properly infer user actions and most effectively deliver perceptually-enabled task guidance.

Figure 1: The EPIC-KITCHENS Dataset.

CCS CONCEPTS

• **Human-centered computing** → **Interaction design**.

KEYWORDS

Egocentric Vision, Action Recognition, Task Guidance

1 INTRODUCTION

Virtual assistants have become more and more ubiquitous as machine learning algorithms have become smarter and computer hardware more powerful. Currently, the most well known virtual assistants rely primarily on audio input to receive commands before transmitting results. Requests are received sequentially and guidance is issued successively, either in the form of audio or visual feedback. In order to be most helpful, however, the ideal virtual assistant would issue on-demand advice based on auditory and visual cues, providing assistance when necessary and remaining silent otherwise. The ultimate goal, as set forth by Draper *et al.*, is a virtual assistant that "sees what you see, hears what you hear, knows your tasks, and knows you. It answers your questions, warns you if you make a mistake, and walks you through unfamiliar procedures" [1]. The ideal virtual assistant is competent and perceptive, providing guidance only when it is capable of transferring knowledge without diminishing user performance.

In addition to perceptive and intuitive, the ideal virtual assistant is capable of delivering guidance through a variety of different methods. Besides providing just-in-time audio feedback, advice may be given by way of instructive visual cues. Checklists, illustrated instructions and instructional videos may all be useful in transferring domain knowledge. However, with the prevalence and advancement of modern augmented reality technologies and computer vision algorithms, visual instructions may be enhanced even further. For instance, instructions may be overlaid onto a user's field of view with the help of an augmented reality headset, placed in relation to the user's environment so that guidance may be interpreted as clearly as possible. Instructions may also be personally calibrated to the user, where advice is given according to how a machine learning model interprets a user's needs and aptitude. Indeed, the ultimate virtual assistant is one that can accurately judge a user's performance and in turn deliver guidance in the most effective way possible.

In order to most successfully perform task guidance, a virtual assistant requires domain knowledge specific to the task at hand. Such knowledge can be established with traditional heuristics, where a virtual assistant is hard coded with prior task knowledge in the form of computer-interpretable task graphs and conditional algorithms. However, to scale most effectively and develop into a true, natural counterpart, the ideal virtual assistant needs to gain knowledge by way of parsing and understanding multi-media instructions. Similar to how many people today prepare for a new task by watching instructional videos, the ideal virtual assistant is capable of gaining knowledge from video and relaying that knowledge in the form of intuitive, non-intrusive advice. By parsing, interpreting and truly learning from multi-media content, a virtual assistant may one day become the perfect sidekick.

2 OVERVIEW

In an effort to assess contemporary technologies in enabling knowledge transfer and perceptually-enabled task guidance, a baseline action recognition model trained on a large egocentric dataset is applied in the context of recipe guidance in the kitchen. Specifically, a baseline Temporal Segment Network [3] trained on the substantial EPIC-KITCHENS dataset [2] is deployed to interpret video instructions and deliver recipe guidance. The network is deployed and evaluated in both offline and real-time scenarios, where offline knowledge gain is used to guide real-time instruction. The

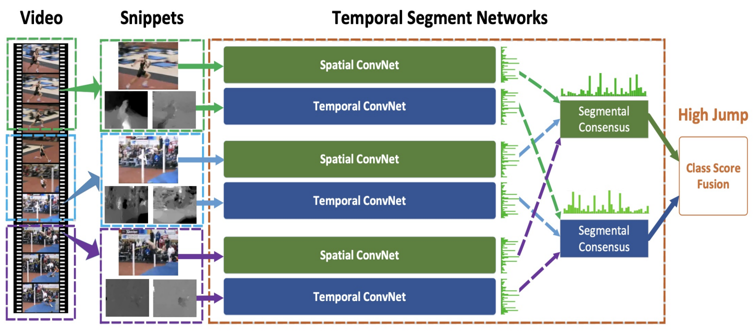


Figure 2: Temporal Segment Network: A video is first split into segments and then snippets before being input into a series of spatial and temporal ConvNets. Predictions from each ConvNet are then fused together by a Segmental Consensus Function to arrive at a final video-level output prediction.

following subsections provide more detail on the overall approach, beginning with an overview of the EPIC-KITCHENS dataset and its role in promoting an accurate and intelligent action recognition system.

2.1 Dataset

The EPIC-KITCHENS dataset, as illustrated in Figure 1, is composed of a large variety of participants interacting in their own kitchen environments. Recorded from the first-person point of view by way of a head mounted GoPro camera, the EPIC-KITCHENS dataset spans 45 kitchens, includes over 100 hours of recording (upwards of 20 million frames) and approximately 90,000 action segments. Recording takes place across 4 cities throughout North America and Europe, offering a collection of highly diverse kitchen interactions and cooking styles. As pointed out by the authors, the EPIC-KITCHENS dataset is "the largest dataset in first-person (egocentric) vision" [2].

During 2020 and 2021, the authors of the EPIC-KITCHENS dataset hosted a series of challenges centered around machine learning and action understanding. Specifically, the EPIC-KITCHENS Challenges seek to identify the most accurate machine learning models in the context of action-recognition, action-anticipation and object-detection. Certifications are awarded to the most accurate classification systems and results are published online. As a starting point, the authors provide a series of baseline neural networks that have been trained on the EPIC-KITCHENS dataset to perform action-recognition. These models were chosen as they demonstrated promising action-recognition performance in the past [2]. One especially promising model is the TSN, or Temporal Segment Network, as introduced by Wang *et al.* [3].

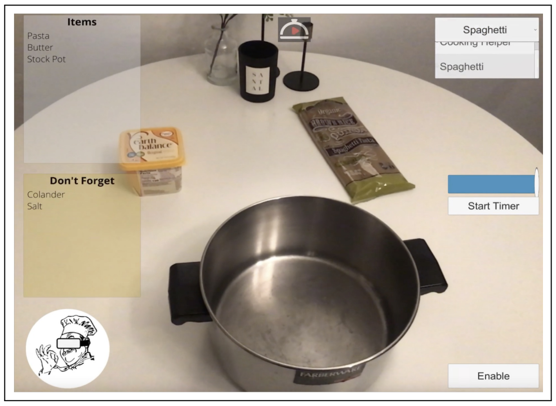
2.2 Temporal Segment Network

The Temporal Segment Network (TSN) is a novel framework developed by Wang *et al.* for video-based action recognition. Built upon the idea of long-range temporal structure modeling, the TSN framework avoids many of the common pitfalls mainstream ConvNet frameworks encounter when performing action recognition. As pointed out by Wang *et al.*, popular ConvNet frameworks oftentimes rely on appearances and short-term motions, thus lacking the ability to either incorporate long-term temporal structure or sufficiently understand a sequence of frames [3]. To avoid such short-term analysis, Wang *et al.* propose a network based on a sparse temporal sampling strategy, where a sparse sampling scheme extracts a diverse array of frames instead of the largely redundant frames acquired when performing dense temporal sampling. With a sparse sampling strategy and very deep ConvNet architecture, Wang *et al.* introduce a network that is capable of capturing relevant information and performing accurate action recognition.

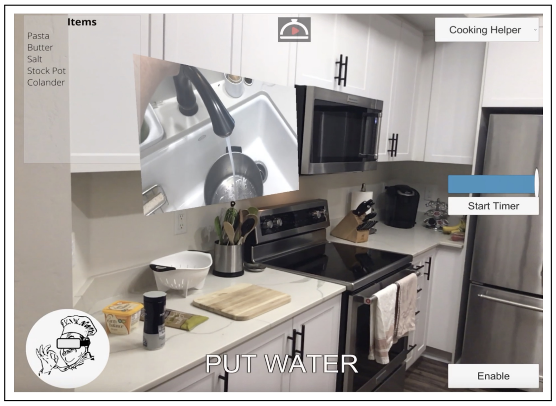
In terms of architecture, the Temporal Segment Network is composed of spatial stream ConvNets and temporal stream ConvNets. Instead of operating on single frames or frame stacks – as is typical of spatial and temporal networks, respectfully – a Temporal Segment Network operates on a sequence of short snippets sparsely sampled from the segments of a video [3]. First considering the snippets independently, the network produces a preliminary prediction for each snippet. Then, in order to arrive at a consensus, the network groups all predictions in a Segmental Consensus Function and outputs a final video-level prediction. Figure 2 illustrates the architecture of a Temporal Segment Network, where a video is first split into segments and then snippets before being classified as an action.

2.3 Video Analysis

Offline and real-time video analysis was conducted in order to determine if a baseline Temporal Segment Network trained on the EPIC-KITCHENS dataset is capable of promoting either knowledge transfer or perceptually-enabled task guidance. In terms of offline analysis, instructional videos are input into the Temporal Segment Network and subdivided into individual clips according to the actions predicted by the model. The predicted action clips are subsequently added to the *AR Cooking Helper* application, where they serve to transfer task knowledge by way of video instruction. In regards to real-time analysis, a trained TSN model is deployed with the *AR Cooking Helper* application to predict a user's actions at runtime. The model's predictions are then used to determine the viability of using a Temporal Segment Network to advance perceptually-enabled task guidance.



(a) Object Recognition – Ingredient and Recipe Recommendations: Upon recognizing tracked kitchen items, the *AR Cooking Helper* issues recipe recommendations (top right drop-down menu) and ingredient recommendations (middle left yellow panel).



(b) Knowledge Transfer – Video Tutorials: Users may utilize a virtual video player to cycle through the steps of a recipe. Displayed recipe steps are chosen during offline video analysis and the corresponding instructions are shown on the bottom of the screen.

Figure 3: The *AR Cooking Helper* utilizes object recognition, textual recommendations, plane tracking and video tutorials to perform task guidance and knowledge transfer.

2.3.1 Offline Video Analysis. Offline instructional videos analyzed by the Temporal Segment Network are split into consecutive, four second long segments. Eight snippets are extracted from each segment, corresponding to a snippet extracted for every 0.5 seconds of segment time. Once gathered, snippets are input into the Temporal Segment Network, where the model classifies the video segment as an action based on a `verb,noun` prediction. Segment predictions are ultimately used to determine which clips of the instructional video are most helpful in facilitating knowledge transfer in the *AR Cooking Helper* application.

2.3.2 Real-Time Video Analysis. During real-time video analysis, image frames are gathered at runtime and action predictions are displayed on screen. Once eight consecutive image frames are collected – where every accessed frame is interspaced by 30 non-accessed frames – the frames are input into a trained Temporal Segment Network as a single video segment. Results output by the model are then used to evaluate model accuracy, infer a user's actions and determine how best to deliver perceptually-enabled task guidance.

3 IMPLEMENTATION

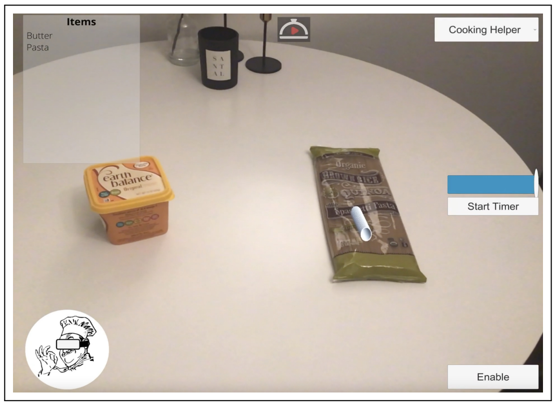
The *AR Cooking Helper* application is implemented on a 7th generation iPad using the Unity game engine [9] and a handful of key augmented reality frameworks. Specifically, to manage augmented reality within the application, the *AR Cooking Helper* relies on Unity's AR Foundation library [10]. AR Foundation is instrumental in supporting the augmented reality tasks plane tracking, anchoring and model placement as well as image tracking and ray casting. Action recognition, on the other hand, is performed by analyzing the device's camera image on the CPU, an image obtained by way of Unity's AR Subsystems library [11]. Once acquired, camera images are collected and then normalized within the application. Upon collecting eight successive camera images, the frames are grouped together in a tensor and input into an Open Neural Network Exchange (ONNX) version of a Temporal Segment Network trained on the EPIC-KITCHENS dataset. Model inference is then performed using Unity's Barracuda inference library [12], with results collected and displayed in the application's *Model Results* pop-up panel.

3.1 Task Guidance

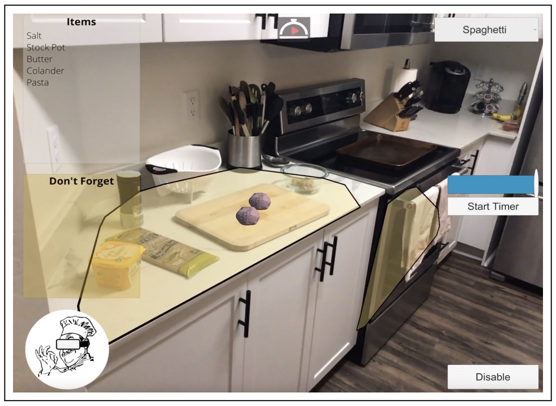
Augmented reality has proven to be a very effective tool in facilitating task guidance in the kitchen [4–8]. In *The AR Cooking Helper*, task guidance is accomplished though a combination of augmented reality, object recognition, textual recommendations, plane tracking and video demonstration. The following subsections reveal how *The AR Cooking Helper* makes use of the user's world space to deliver task guidance and cultivate knowledge transfer.

3.1.1 Recommendations. The *AR Cooking Helper* relies on AR Foundation's Image Tracking Library to perform object recognition and deliver suitable recipe recommendations. Serving as a repository of recognized kitchen items, the Image Tracking Library detects ingredients in view at runtime and determines the type and timing of task recommendations. Upon detecting the ingredients *stock pot* and *pasta*, for example, the *AR Cooking Helper* provides a new recipe recommendation (spaghetti) and a list of recommended ingredients (salt, butter, etc.). Ingredient suggestions are then updated in real-time as new items are recognized by the Image Tracking Library. Figure 3a provides an illustration of the *AR Cooking Helper* issuing ingredient and recipe recommendations after the system has recognized the items *pasta*, *butter* and *stock pot.*

3.1.2 Video Demonstration. To walk a user through a recipe, the *AR Cooking Helper* provides the set of instructional video clips prepared during offline video analysis. All video clips are played through the application's video player, or a two dimensional plane that is virtually rendered in the user's world space. The user may place the

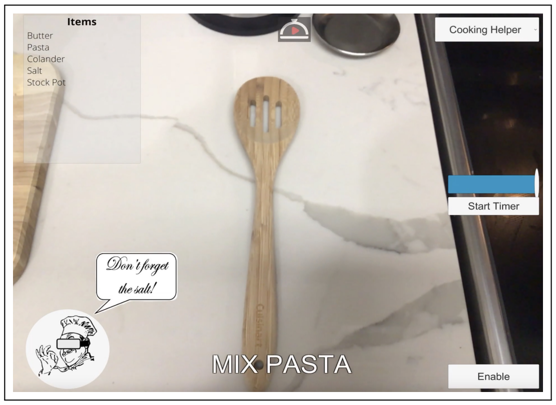


(a) Virtual Models – Object Tracking: Virtual models are rendered on top of tracked objects to indicate when key items have been identified by the *AR Cookin Helper*. Here, a virtual noodle is spawned on top of the tracked object *pasta*.

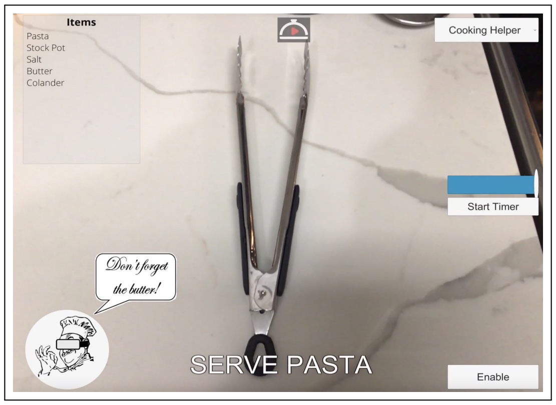


(b) Virtual Models – Item Templates: Virtual models provide the user with points of comparison when cooking. Virtual meatballs are included in the Spaghetti recipe to assist the user in making consistently sized meatballs.

Figure 4: Virtual models in the *AR Cooking Helper* provide both helpful feedback and instructive points of comparison.



(a) Recognizing the item *wooden spoon* before the item *salt* suggests the user performed the action *Mix Pasta* before *Add Salt*.



(b) Recognizing the item *tongs* before the item *butter lid* suggests the user performed the action *Serve Pasta* before *Add Butter*.

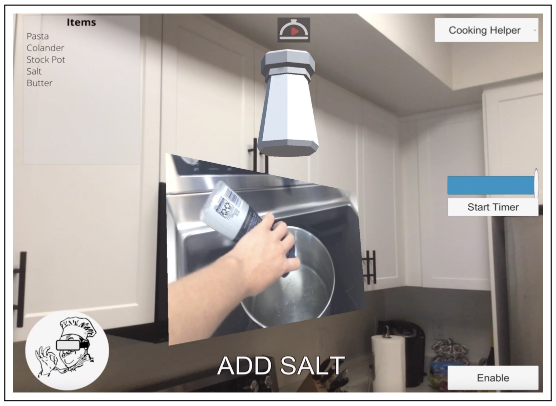
Figure 5: The *AR Cooking Helper* keeps track of what objects are recognized when to infer the user's progress in a recipe. When a future object is recognized before an anticipated object, the application assumes recipe steps have been performed out of order and issues a corrective warning in the bottom left corner of the screen.

video player on either vertical or horizontal surfaces, so long as the surface is recognized by AR Foundation's AR Plane Manager. Once placed upon a valid plane, the user may cycle through the steps of the recipe, where the instructions corresponding to each recipe step are displayed on the bottom of the screen. Figure 3b provides an example of the instructional video player and the instructions associated with the selected recipe step.

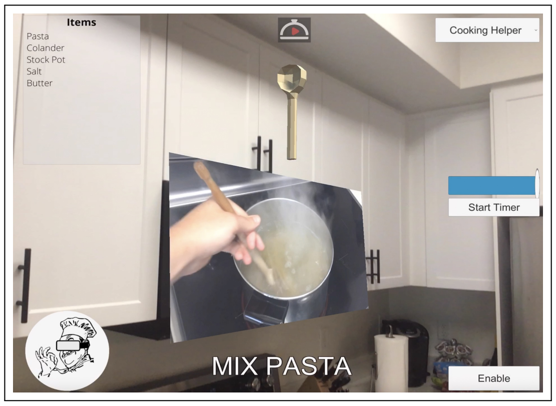
3.1.3 Virtual Models. Virtual models are used extensively in the *AR Cooking Helper* to provide both helpful feedback and virtual points of comparison. When the Image Tracking Library identifies a tracked object, for example, a virtual model is rendered above the kitchen item to clearly indicate that the system has successfully performed object recognition. When a user is working through a recipe, virtual items are also dispensed with proper user input; virtual meatballs associated with the Spaghetti recipe, for example, are scaled and spawned in the user's world space when the user employs touch gestures on top of a tracked plane. Figure 4 demonstrates virtual model placement, where a virtual noodle is rendered on top of the tracked object *pasta* and virtual meatballs are rendered on a countertop to provide the user with a size template when cooking spaghetti.

3.2 Mistake Recognition

In addition to employing the Image Tracking Library for object recognition, the *AR Cooking Helper* utilizes tracked images to discern user action. By keeping track of what objects are identified



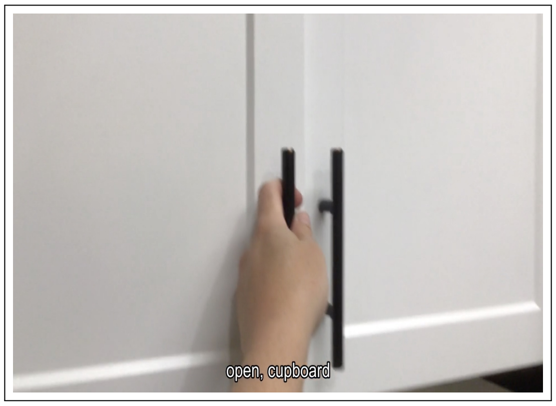
(a) A virtual salt shaker is rendered above the video player when the user gazes at the video player during the *Add Salt* step.



(b) A virtual wooden spoon animates above the video player when the user gazes at the video player during the *Mix Pasta* step.

Figure 6: The *AR Cooking Helper* renders virtual kitchen items associated with the current recipe step when the user gazes at the video player. Virtual items provide additional task context should the user struggle with the video player instructions.

(a) Accurate action prediction output by the TSN when analyzing an offline video from the EPIC-KITCHENS dataset.



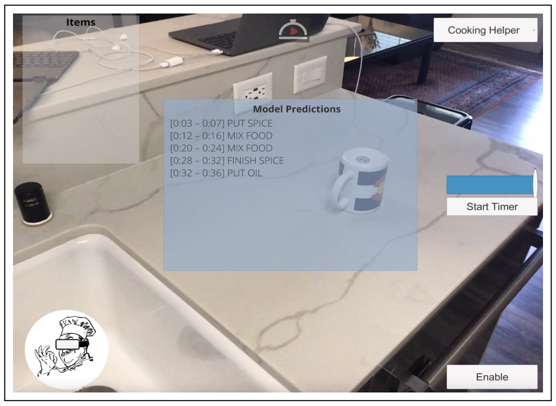
(b) Accurate action prediction output by the TSN when analyzing an offline video recorded in my apartment kitchen.

Figure 7: A Temporal Segment Network was evaluated using offline videos acquired from the EPIC-KITCHENS dataset and my own home recordings. Action predictions are made for every segment of a video and then added back to the video as subtitles.

when, the *AR Cooking Helper* is able to infer a user's progress in a recipe. Noticing the item *wooden spoon* before the item *salt*, for example, suggests the user has performed the action *Mix Pasta* before *Add Salt*. Likewise, recognizing the item *tongs* before the item *butter lid* indicates that the user skipped the step *Add Butter* before performing *Serve Pasta*. When mistakes are recognized, the *AR Cooking Helper* issues a helpful warning, allowing the user to stay on track with the steps of the recipe. Figure 5 illustrates the warnings issued by the *AR Cooking Helper* when the application realizes the user has performed the steps of the Spaghetti recipe out of order.

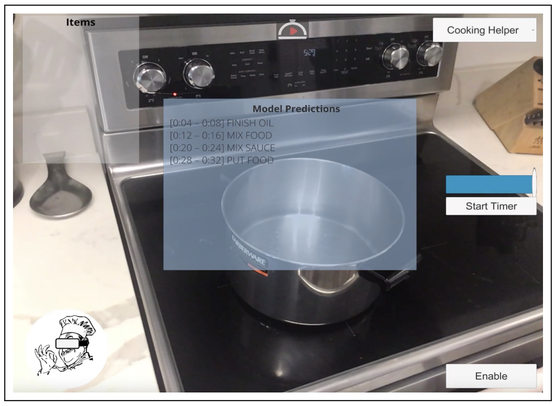
3.3 Gaze Interaction

Besides using object recognition to infer user action, the *AR Cooking Helper* also incorporates gaze interaction to deduce a user's actions and task performance. If a user is stuck on a certain recipe step, for instance, it is assumed that they will linger in front of the instructional video player, continually replaying the step's video instructions in order to fully understand the task at hand. Upon recognizing the user's gaze, the *AR Cooking Helper* will render a virtual model associated with the recipe step, using the opportunity to further clarify the recipe's instructions and highlight key items. To most effectively deliver the supplementary guidance, the virtual kitchen item is rendered on top of the video player, where it repeats a simple animation in an attempt to catch the user's attention and convey additional information. Figure 6 provides an example of how the *AR Cooking Helper* uses gaze interaction to infer a user's task performance and deliver supplementary task guidance.



```
Model Predictions
[0:03 – 0:07] PUT SPICE
[0:12 – 0:16] MIX FOOD
[0:20 – 0:24] MIX FOOD
[0:28 – 0:32] FINISH SPICE
[0:32 – 0:36] PUT OIL
```

(a) Real-Time Video Analysis – Cleaning: The TSN output illogical action predictions when analyzing the task *clean cup*.



```
Model Predictions
[0:04 – 0:08] FINISH OIL
[0:12 – 0:16] MIX FOOD
[0:20 – 0:24] MIX SAUCE
[0:28 – 0:32] PUT FOOD
```

(b) Real-Time Video Analysis – Cooking: The TSN output inaccurate action predictions when analyzing the Spaghetti recipe step *Add Salt*.

Figure 8: Real-time action recognition is performed within the *AR Cooking Helper* by utilizing a trained Temporal Segment Network formatted in the ONNX file standard. Actions are analyzed every four seconds and displayed within the application's pop-up panel.

4 EVALUATION

In order to determine the practicality of relying on a Temporal Segment Network to achieve perceptually-enabled task guidance and knowledge transfer, a series of experiments were conducted in both offline and real-time environments. To measure the generalizability of the model, the network is input an array of video segments from the EPIC-KITCHENS dataset as well as video segments recorded in my own home kitchen. The following subsections examine the model's performance in a variety of settings and indicate whether or not a Temporal Segment Network is well suited to support the ideal virtual assistant.

4.1 Offline Video Analysis

The Temporal Segment Network performed surprisingly well when analyzing offline videos acquired from either the EPIC-KITCHENS dataset or my own home video collection. In both environments, the trained model output logical `verb,noun` predictions for each action segment of the video. Interestingly, the model was more accurate when classifying cleaning tasks rather than cooking tasks, an admittedly disappointing finding considering the model's application in the *AR Cooking Helper*. Nevertheless, the Temporal Segment Network performed well enough in an offline setting to justify further analysis of the network in a real-time setting. Figure 7 gives an example of the model's performance when predicting the egocentric kitchen interactions found in either an EPIC-KITCHENS video or my own home video.

4.2 Real-Time Video Analysis

Unlike during offline analysis, the Temporal Segment Network performed poorly when attempting to classify actions in a real-time setting. Analyzing kitchen interactions that included both cooking and cleaning, the model consistently output illogical predictions that did not match the user's current task. Taking into account the differences between offline and real-time analysis, the Temporal Segment Network may perform poorly when deployed within the *AR Cooking Helper* due to the Barracuda library requirement that the model be formatted in the ONNX file standard. Whatever the cause, the real-time experiments clearly indicate that a significant amount of model improvement is necessary to accurately inform the virtual assistant and advance perceptually-enabled task guidance. Figure 8 demonstrates the inaccurate action predictions made by the model when analyzing both cooking and cleaning tasks in my home kitchen.

4.3 Model Analysis

A series of experiments were conducted in an effort to explain why the Temporal Segment Network performs poorly in a real-time setting yet accurately in an offline setting. Specifically, the model was evaluated in real-time using a combination of normalization techniques and inference engines, where the best combination would ideally produce accurate action predictions similar to those produced during offline analysis.

As pointed out by the authors of the Barracuda inference library, model performance may vary depending on which core engine is used to execute the network; a `CSharpBurst` backend may perform inference the fastest but may not deliver as accurate results as the more stable `CSharpRef` backend [12]. Data normalization may also play a role in model performance. To measure the effect of data normalization, the pixel values of each video frame are normalized according to either a *Standard* strategy (each color channel is normalized to have a standard deviation of 1 and a mean of 0) or a *TSN* strategy (each color channel is normalized based on the values used by the TSN authors during normalization). Table 1 provides a break down of the different combinations used when evaluating the Temporal Segment Network in real-time, where no combination produced accurate action predictions like those produced during offline analysis.

Core Engine	Normalization
CSharpRef	Standard
CSharp	Standard
CSharpBurst	Standard
CSharpRef	TSN
CSharp	TSN
CSharpBurst	TSN

Table 1: Real-Time Analysis – Evaluating real-time model performance using a variety of inference engines and normalization strategies. No combination produced accurate action predictions like those output during offline analysis (Core Engine: CSharpRef, Normalization: TSN).

5 FUTURE WORK

As mentioned above, more work needs to be done on the Temporal Segment Network to produce a more accurate and adaptive model. Indeed, in order to properly support a virtual assistant, an action recognition model must be able to accurately analyze both offline instructional videos and real-time user actions. Iterative experiments evaluating model performance under a variety of video segmentation strategies may yield more accurate results. Additionally, fine-tuning the model's consensus function may lead to logical action predictions more closely aligned with the user's actions in the kitchen. Whatever the approach, a virtual assistant needs to be able to rely on an accurate action recognition model in order to develop into a capable and useful counterpart.

In terms of design, *The AR Cooking Helper* may become an even more effective augmented reality application by incorporating a few interface adjustments. For one, user studies revealed that the application's drop-down menus and panels were inconspicuous; rather than relying on menu animations and textual cues to capture user attention, feedback should be delivered primarily by virtual models placed clearly in the user's world space. Additional auditory cues would also benefit the application, providing the user with a more instructive and engaging experience. Finally, to make for the best cooking experience, *The AR Cooking Helper* should be implemented on a hands-free device – such as a Microsoft HoloLens or Oculus headset – thereby allowing the user to simultaneously cook with both hands and receive non-intrusive task guidance while in the kitchen.

6 CONCLUSION

In this research project, a Temporal Segment Network trained on the EPIC-KITCHENS dataset was deployed in an augmented reality application to assess the effectiveness of a baseline action recognition model in promoting perceptually-enabled task guidance and knowledge transfer. By investigating model performance in offline and real-time settings – as well as exploring techniques in image tracking and gaze interaction – the project shed light on the most promising approaches to take when striving to infer egocentric user action and develop the ideal virtual assistant.

REFERENCES

[1] Bruce Darper, Bill Bartko, Gretchen Eitt and Alex Goldberg. 2011. DARPA, Founders Day: Perceptually-enabled Task Guidance (PTG).

[2] Dima Damen and Hazel Doughty and Giovanni Maria Farinella and Sanja Fidler and Antonino Furnari and Evangelos Kazakos and Davide Moltisanti and Jonathan Munro and Toby Perrett and Will Price and Michael Wray. 2018. Scaling Egocentric Vision: The EPIC-KITCHENS Dataset.

[3] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. 2016. ECCV, Amsterdam, Netherlands. Temporal Segment Networks: Towards Good Practices for Deep Action Recognition.

[4] Leonardo Bonanni, Chia-Hsun Lee, and Ted Selker. 2005. A framework for designing intelligent task-oriented augmented reality user interfaces. In Proceedings of the 10th international conference on Intelligent user interfaces (IUI '05). Association for Computing Machinery, New York, NY, USA, 317–319. DOI:https://doi.org/10.1145/1040830.1040913

[5] Leonardo Bonanni, Chia-Hsun Lee, and Ted Selker. 2005. Attention-based design of augmented reality interfaces. In CHI '05 Extended Abstracts on Human Factors in Computing Systems (CHI EA '05). Association for Computing Machinery, New York, NY, USA, 1228–1231. DOI:https://doi.org/10.1145/1056808.1056883

[6] Leonardo Bonanni and Chia-Hsun Lee. 2004. The Kitchen as a Graphical User Interface. In ACM SIGGRAPH 2004 Art gallery (SIGGRAPH '04). Association for Computing Machinery, New York, NY, USA, 109–111. DOI:https://doi.org/10.1145/1185884.1185989

[7] Wendy Ju, Rebecca Hurwitz, Tilke Judd, and Bonny Lee. 2001. CounterActive: an interactive cookbook for the kitchen counter. In CHI '01 Extended Abstracts on Human Factors in Computing Systems (CHI EA '01). Association for Computing Machinery, New York, NY, USA, 269–270. DOI:https://doi.org/10.1145/634067.634227

[8] Halupka V., Almahr A., Pan Y., Cheok A.D. (2012) Chop Chop: A Sound Augmented Kitchen Prototype. In: Nijholt A., Romão T., Reidsma D. (eds) Advances in Computer Entertainment. ACE 2012. Lecture Notes in Computer Science, vol 7624. Springer, Berlin, Heidelberg. DOI:https://doi.org/10.1007/978-3-642-34292-9_43

[9] Unity: Real-Time Development Platform, `https://unity.com/`

[10] Unity: AR Foundation, `https://unity.com/unity/features/arfoundation`

[11] Unity: AR Subystems, `https://docs.unity3d.com/Packages/com.unity.xr.arsubsystems@4.1`

[12] Unity: Barracuda, `https://github.com/Unity-Technologies/barracuda-release`