RESEARCH ARTICLE

A Review on Deep-Learning Based Egocentric Action Anticipation

Richard Wardle[1], Sareh Rowlands[1]

[1] University of Exeter, Exeter, United Kingdom

*Corresponding author: Sareh Rowlands: s.rowlands@exeter.ac.uk

Citation: Wardle R., Rowlands S. (2025) A Review on Deep-Learning Based Egocentric Action Anticipation. Open Science Journal 10(1)

Received: 6th September 2024

Accepted: 17th December 2024

Published: 20th January 2025

Funding: The author(s) received no specific funding for this work

Competing Interests: The authors have declared that no competing interests exists.

Abstract:

As autonomous systems become more embedded into our environments, the ability of these systems to anticipate the future actions of humans will become invaluable for providing assistance and safety measures. Egocentric action anticipation is a task in which a future activity must be predicted using first-person footage. This project is a survey that aims to provide an updated view of advancements within this task, to guide architecture design for future implementations. This survey has chosen a range of publicly available egocentric action anticipation models.

Keywords: Action anticipation, Egocentric vision, Deep learning, Transformers

Introduction

Ubiquitous automation is becoming an unavoidable reality, as technology and machines continue to integrate seamlessly into everyday environments. To function effectively, these systems must possess the ability to anticipate the actions of external agents, such as humans, in real time. Action anticipation is a crucial task that enables intelligent systems to predict future actions based on observed patterns, allowing them to plan and respond optimally. This capability holds significant promise for enhancing automation across various domains, from robotics and autonomous vehicles to assistive technologies. Researchers have formalised action anticipation as a task that involves providing a model with a video segment prior to the moment of prediction. Using this input, the model must infer the action class labels for a future segment of the video. To achieve this, the model generates a multi-modal distribution of predictions, which provides a probabilistic estimation of potential actions. The time interval between the observed segment and the predicted actions can vary; in this study, it ranges between 0.25 seconds and 2 seconds, offering short-term forecasts that are practical for real-world applications.

This study specifically focuses on egocentric action anticipation, where predictions are made using footage captured from a first-person perspective. This approach is particularly relevant for wearable devices, such as augmented reality (AR) glasses, which are equipped with first-person cameras and can deliver real-time, assistive feedback to users. In industrial environments, for instance, AR glasses could anticipate a worker's next action and provide relevant instructions, improving efficiency, safety, and task execution. Such applications highlight the

immense practical potential of egocentric action anticipation technology in augmenting human capabilities and supporting intelligent, responsive automation.

The paper provides a detailed examination of the current state of egocentric action anticipation, using the results of the Epic-Kitchens challenge—a benchmark competition for evaluating predictive models. By analysing the strengths and weaknesses of the most recent predictive architectures, this study identifies key trends and areas for improvement in the field. It highlights the growing dominance of self-attention-based models, such as transformers, which excel at processing sequential data and learning dependencies across video frames. The insights gained from this analysis help to establish a clearer direction for advancing the task, identifying the most effective methods for achieving accurate and generalisable action predictions. The novelty and significance of this work lie in its comprehensive evaluation of state-of-the-art approaches, offering a thorough understanding of the current landscape of egocentric action anticipation. By identifying gaps and opportunities for improvement, this study serves as a valuable foundation for future research and innovation. The findings have far-reaching implications, particularly for real-world applications in AR technology, where intelligent systems capable of anticipating human actions can fundamentally transform industries, improve user experience, and enhance automation in dynamic environments.

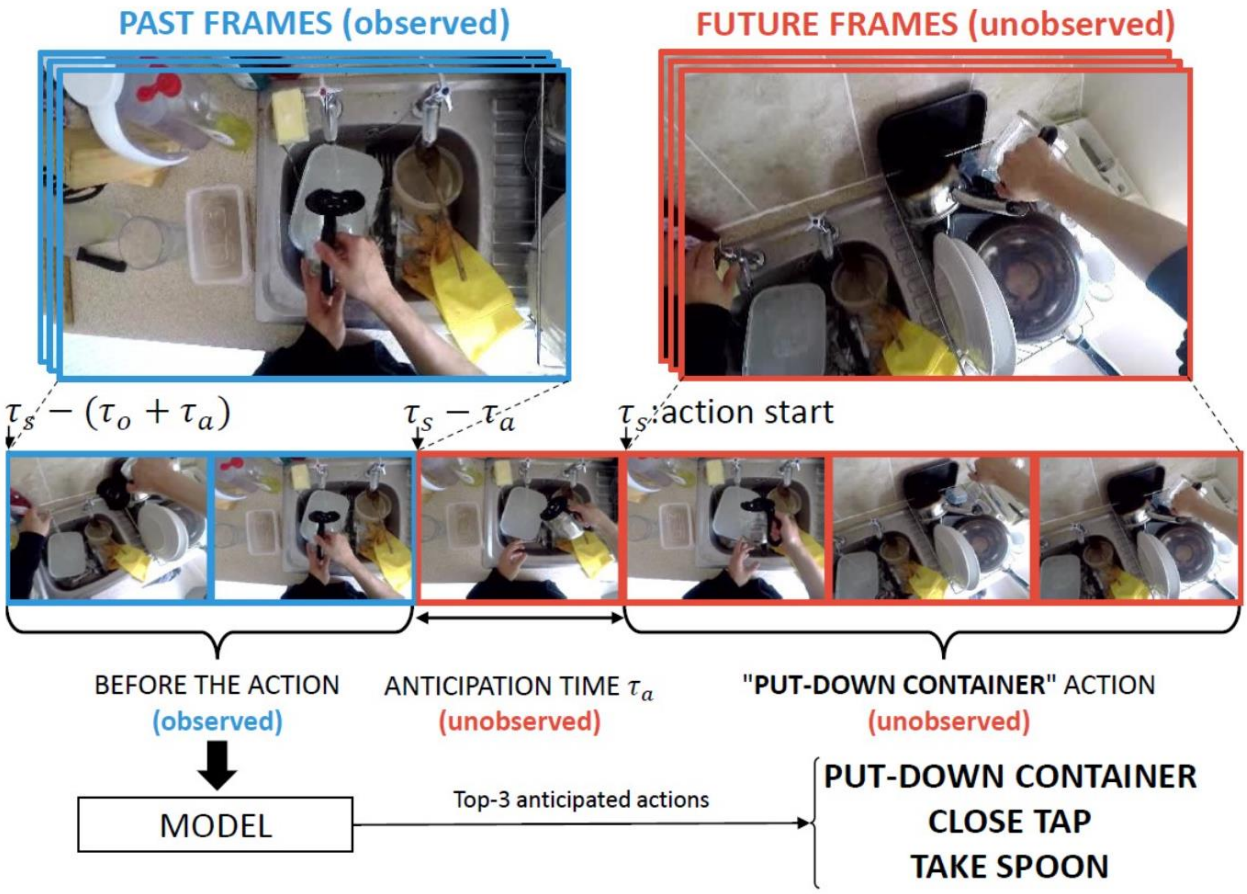

Figure 1. Egocentric Anticipation Task using Top-3 Actions [1]

Background

This section discusses various approaches to egocentric action anticipation, highlighting the use of different neural network architectures for predicting future actions. Feature extraction involves RGB video, object detection, and optical flow to

capture relevant motion and object information. CNNs process spatial features, but lack temporal persistence, while RNNs and LSTMs overcome this by modelling sequential dependencies, with LSTMs improving on the vanishing gradient problem. GRUs offer simpler alternatives with reduced training times, though they share limitations with LSTMs. Finally, transformers enhance action anticipation by processing sequences in parallel using a self-attention mechanism, outperforming other models in this domain. This section will also cover the critical information from the literature review and recent mention of novel techniques and results.

Features and extraction

Generally, datasets for egocentric action anticipation consist of RGB video. This data is then directly fed into an end-to-end model or is pre-processed to extract further modalities that focus on a video, such as a hand mask or object detection. Aside from RGB, the most typically used modality for this select task is object detection [2]. Objects actively used within a scene are identified; this assists the model by identifying objects that can assist identification of target variables for the prediction task (noun and action object pairing labels). The Object modality is extracted using a Faster R-CNN, which results in vectors summarising object classes within the current frame. Optical flow represents the changes of patterns between frames, which within egocentric action anticipation can be the result of head movement, object movement or hand/arm movement. This is useful in analysing motion. Optical flow is extracted from RGB using a temporal segment network (TSN) and pre-trained for an action recognition task.

Convolutional neural networks

A convolutional neural network (CNN) works by gathering and processing information within a grid structure. CNN works by creating a filter that slides along every possible frame position. CNNs do have inherent issues within the architecture, namely persistence. This network has no mechanism to inform future predictions based on previous outputs. This means that CNNs can only work over static frames but can be used for video processing when combined with a network that can process temporal information.

Furnari et al. [3] uses two feed-forward 3D CNNs combined with a graph to transfer knowledge between the action recognition network and the action anticipation network. However, this method under-performed the baseline model of rolling-unrolling LSTM (RULSTM) and was over two times faster than the recurrent-based network in training and inference time. A multi-modal temporal CNN [4] prioritises inference time within its design. This model achieved greater accuracy than the RULSTM baseline, with inference time still twice as fast.

Recurrent neural networks

Recurrent Neural Networks (RNNs) allow for information persistence - which is not viable within CNNs. This property is gained using a looping mechanism that provides information (a representation of previous inputs known as the hidden state) to flow between each computational step. The hidden state is fed forward through the network allowing for sequential memory and pattern identification over a sequence of information. In terms of action anticipation, RNNs can detect patterns over time. When the network updates weights during gradient descent, issues lie in how the weight that connects the hidden layer is updated. These weights experience an exploding or vanishing gradient.

Long-short term memory

Long-Short Term Memory networks (LSTMs) [5] are a recurrent-based architecture designed to improve RNNs by allowing for long-term dependencies while eliminating the risk of vanishing gradients. LSTMs gated architecture has demonstrated superiority against the standard tanh module RNN within parameter-rich data sets [6]. Panasonic's "CNSIC PSNRD" [7] is the current best-performing LSTM-based method, coming second place within the Epic-Kitchens 2021 challenge. The implementation is based on the Rolling-Unrolling LSTM but with further architectural optimisations including label smoothing, an uncertainty-based loss function and test-time augmentation. Limitations of LSTMs, which are highlighted during the action anticipation task, is that every frame of data is not rich with features that will correlate with the upcoming activity, which introduces noise that standard LSTMs cannot eliminate.

Gated recurrent unit

Gated Recurrent Unit (GRU) is the most recent type of recurrent architecture proceeding the LSTM. While similar, the main difference is the simplicity of each cell structure. This structure is particularly advantageous in reducing training times by increasing efficiency due to fewer network operations. Additionally, the network can train on less training data, but the fewer parameters also mean a shorter memory than the LSTM. Huang et al. [8] utilise a GRU within a graph structure to compare context relationships between events and the related features. This model outperformed many implementations, including the RULSTM and ImagineRNN (included in this survey). Limitations of GRUs are similar to LSTMs, in that every frame of data is not correlated with the upcoming activity, so it introduces noise that cannot be eliminated.

Transformer

Transformers improve on weaknesses of recurrent based modules like the LSTM and GRU [9]. This weakness is based on the recurrent nature of the models - they cannot be run in parallel due to needing to process information sequentially. While this option is available in CNN- based implementations, the supplementing networks become too large to be helpful within this application. The transformer is considered to have three main characteristics. Firstly, it processes information as a whole and in parallel and not sequentially like a recurrent network. The self-attention mechanism of a transformer allows for every part of the input sequence to be modelled for dependencies against every other part of the sequence, which allows the net- work to learn what is important within the data. Finally, as there is no recursion mechanism, sequences are encoded with their positions related to their place in the input sequence. This architecture not only improves on training times of RNN but has been shown to outperform all current existing architectures in action anticipation.

The Data

The Epic-Kitchens [10] dataset used within this project contains 55 hours of first-person recordings from a head-mounted Go-Pro that captures a variety of tasks within 32 separate kitchens. The dataset contains 39596 action annotations, 125 verbs, 331 nouns and 2513 actions. The test set for this data contains two distinct collections of seen and unseen kitchens to ensure models are not over-fitted to the training sets and allow for a measure of model generalisability. The seen kitchens set contains 32 kitchens within the training and testing sets, with a split of 80% training and 20% testing. The unseen data is split into whole sequences, meaning each kitchen is either in the training or test set, never within both. 28 kitchens are

allocated to the training set, leaving 4 kitchens seen in the test set, giving a 93% training and 7% test split. EGTEA Gaze+ [11] contains a semi-scripted dataset containing 28 hours worth of cooking activities filmed in the first-person perspective from 32 individuals in different kitchens. It includes 19 verbs, 51 nouns and 106 individual actions. The data is split into three sets of training and validation sets.

The Models

This section introduces the models that this survey aims to analyse. Backgrounds into each model implementation will provide a framework for understanding the results and comparing each method in the analysis section.

Rolling-Unrolling LSTM

The Rolling-Unrolling LSTM (RULSTM) [1] is an RNN based architecture. RULSTM model works on three modalities, RGB, optical flow and object detection. These modalities are pre-extracted, as mentioned within section 3. The overall architecture is based on the completion of two distinct sub-tasks. The encoding module (Rolling-LSTM) works on past observations, attempting to summarise past events until a targeted time step.

The Unrolling-LSTM attempts to use cell vectors and hidden states from the Rolling-LSTM to anticipate target action. Initially, the two modules undergo pre-training using a novel architecture: Sequence Completion Pretraining (SCP). SCP modifies each module's internal connections to specialise in the desired sub-task. During the process of SCP, the Unrolling-LSTM internal states are computed by using sample input representations from future time-steps [1].

After encoding is completed, the hidden and cell vectors are passed into the U-LSTM. The ULSTM then iterates on the video clip (fm,t) (nt) times until the start of the target action is reached. Hidden and cell states for the U-LSTM are computed at each iteration. This structure is repeated within every branch of the model. Action scores are then computed using the last hidden vector of each branch's U-LSTM using linear transformation based on learnable parameters [1]. To ensure information from functional modalities is preserved, the modality attention module (MATT) computes an attention score for each modality that indicates the relative importance within the final prediction [1].

ImagineRNN

ImagineRNN [12], uses the aforementioned RULSTM as a base architecture and performs the task using a recurrent neural network architecture. To optimise the model, ImagineRNN is asked to pick the correct future states from a series of "distractors". This contrastive learning task ensures that a trained ImagineRNN can detect the change of action states at different times. The model has additional supervision by using future intentions. This attempts to extract the purpose for the actions in the current frame, which helps to predict the following future action. The issue in this approach is that adjacent frames are likely to share similar visual features, so model capacity is wasted in predicting the future intention for this case. Instead, the network is trained to forecast future changes instead of the future frame features.

Latent goal

Latent Goal bases its architecture on the supposition that, for any goal, a series of actions are completed to reach an implicit goal. The action anticipation task would greatly assist in extracting the inherent goal that the agent is attempting to accomplish. Latent Goal Learning [13] aims to compute the goal from within a clip

to anticipate the following action. The implementation involves using a series of stacked LSTMs (based on the average number of actions per video clip). Each LSTM unit represents intermediate actions leading to the latent goal. These LSTMs provide the latent goal, of which the proceeding visual representation should be closer to the latent goal than the embedded visual representations. To encourage separation, a threshold is given between the latent goal and the observed video representations.

A single LSTM is used to iterate through all candidate actions, determining their potential as the following action. The most likely candidate is what the current video representation calculated to be closest to the latent goal. Three criteria assist this process. First, the following visual feature is calculated using the sequence goal's action validation and observed representation. Then, the visual feature is compared with the latent goal, yielding the candidate closeness, which measures if the visual feature is closer to the goal. Finally, the initial latent goal is compared with the newly generated latent goal. This measurement yields the distance between the latent goal and the anticipated action. The action candidate is selected from these criteria, and the process is repeated from the next action in the sequence.

Self-regulated learning

Self-Regulated Learning [14] focuses on extracting context from the video clips. This is done by looking for contextual relations along with temporal range: looking at extracting these relations over both the features and semantic information from within the data. LR works by using a three-stage architecture. Firstly, features are extracted using a TSN and then aggregated to encode the observed video representations, which yield a hidden representation that is passed into the recursive sequence. The second step uses a series of GRUs that predict the following time step, feeding the previously generated hidden state and video representation until the desired time step is reached. Finally, a linear layer with a soft-max activation function is used to create a probability distribution for the target activity at the final time-step.

Future prediction works by extracting features from an observed clip into feature representations, which are then recursively run to produce the next prediction. This process continues until the target time is reached. The semantic context describes the relationship between activities and objects in a target activity. To aid in extracting the semantic context, two additional tasks of action classification and object categorisation are run under a multi-task learning framework, improving performance on the main task. This helps the network anticipate future actions, as the verb and noun are target labels for the task.

Temporal aggregation

Temporal Aggregation [15], uses an attention-based architecture for predictive modelling. This improves on the limitations of RNNs bottleneck of only working well on small sequences (due to the tendency to pay more attention to the final parts of a sequence from being fed sequentially). Attention introduces a form of memory within the network - the attention architecture in Temporal Aggregation allows for storage of attention weights throughout all inputs into the network giving context to all frames [16]. This implementation uses optimisation techniques such as max-pooling and a novel representation of processing inputs. Additionally, temporal Aggregation works by using an ensemble of techniques that use the long-range past. This model's representation focuses on extracting long-range observations and recent representations at various levels.

To reduce the computational load of pooling low and high-level features, concatenating max-pooled features between two frames creates a snippet. This method is later shown to be effective at the task while remaining lightweight over large video clips. Recent and spanning features are extracted using start and end

frames and a number of intermediate snippets. Recent features are defined as a feature bank of snippet features with different start frames. Spanning features is a feature bank of snippet features with a varying number of snippets. These feature snippets are combined as an ensemble by varying the number of snippets from the spanning past and keeping the number of features from the recent past constant. To capture relationships among and between spanning and recent snippets, spanning blocks (derived from non-local blocks) are created. Spanning blocks compute fixed-length representations for snippets. Outputs from multiple coupling blocks are concatenated to produce aggregated temporal representations of recent and long-range past. To anticipate future action, multi-step estimates are computed. All temporal aggregates and classification layer outputs are run through a linear layer. The output of which is fed into a single-layer LSTM. The LSTM predicts action and duration vectors for each time step. Dense anticipation loss sums the cross-entropy over the current action, current action duration, future actions and future action duration. This process is applied recursively until the target time step is reached.

AVT

The adaptive video transformer (AVT) [17] is an end-to-end model based on a two-stage architecture. A notable feature is the casual attention modelling, which only predicts feature actions based on the observed frames. The AVT is the first end-to-end transformer architecture for video and uses self-attention for high-level reasoning and image recognition. Like Temporal Aggregation, attention is leveraged; however, self-attention can model dependencies between varying parts of the input sequence leading to further extraction of relationships. This provides a significant degree of generalisability as every input feature attends to every other input feature allowing for the extraction of relationships between all features in a sequence [16]. The backbone network of the AVT (AVT-b) is based on a vision transformer (ViT). This network is attention-based, extracting a feature representation for each frame. Future features are predicted using the head of the network (AVTh); this works by applying a Casual Transformer Decoder, the output of which is decoded into a distribution of predictions. This distribution is then mapped over the semantic action classes of which the final prediction is extracted. Training of AVT is supervised with three separate losses. A cross-entropy loss is used to supervise the next action prediction with future labels. The focus is on intermediate predictions at the feature level; future features are predicted to match actual features that appear within the clip using self-supervision. To assist prediction within the action class, action labels from the dataset are used to supervise intermediate predictions (when a clip overlaps with labelled action segments preceding the target segment).

Evaluation metrics

Top-1 Accuracy: The most intuitive of the evaluation metrics is Top-1 accuracy. Accuracy is defined by comparing the model's highest-ranked prediction of a verb, noun and action to the ground truth labels.

Top-5 Accuracy: Top 5 Accuracy takes the top 5 highest scored anticipatory predictions from the model and the predictions' accuracy against the ground truth. Accuracy of Top-5 over Top-1 accuracy has been demonstrated experimentally to effectively recover true rankings in multi-label learning algorithms.

Visibility of the tests

The testing will be based on the structure of seen and unseen test sets to measure each model's generalisability towards novel environments. The seen test set will include all the identical kitchens between training and testing (28 in total). The sequences within the footage are split 80% into training and 20% into testing. So while the identical kitchens are within seen, no same sequence is used. Unseen is

divided so all clips for individual kitchens are split into training or testing, with four new kitchens and participants presented. 7% of the frames within the total dataset are unseen. This testing implementation was created within the first Epic-Kitchens dataset [18].

Table 1 Comparing Seen and Unseen test splits [18]

Subjects	Sequences	Duration(s)	Action Segments
Train/Val	28	272	141731	28,561
S1 Test	28	106	39084	8,064
S2 Test	4	54	13231	2,939

Varying anticipation times

Within the formal training and testing, all models are trained to anticipate with a time step of 1 second ahead of the shown clip. To assess how adaptive the models can be with their predictions, anticipation times will be iterated from 0.25 seconds to 2.0 seconds (using 0.25- second increments).

Results

The results are presented in the following tables:

Table 2 Top-1 Accuracy on Epic-Kitchens-55

Top-1 Accuracy %	Seen Kitchens			Unseen Kitchens		
Model	Verb	Noun	Action	Verb	Noun	Action
RULSTM	33.04	22.78	14.39	27.01	15.19	8.16
ImagineRNN	35.44	22.79	14.66	29.33	15.50	9.25
Self-Regulated	34.86	22.83	14.22	27.42	15.47	8.81
TemporalAgg	37.87	24.10	16.64	29.50	16.52	10.04
AVT+	34.37	20.16	16.84	30.65	15.64	10.40
LatentGoal	27.96	27.40	8.10	22.41	9.23	4.78

Table 3 Top-5 Accuracy on Epic-Kitchens-55

Top-5 Accuracy %	Seen Kitchens			Unseen Kitchens		
Model	Verb	Noun	Action	Verb	Noun	Action
RULSTM	79.55	50.95	33.73	69.55	34.38	21.10
ImagineRNN	79.72	52.09	34.98	70.67	35.78	22.19
Self-Regulated	79.57	52.05	34.59	71.90	36.81	22.03
TemporalAgg	79.74	53.98	36.06	70.13	37.83	23.42
AVT+	80.02	51.57	36.52	72.17	40.76	24.26
LatentGoal	78.09	55.98	26.46	71.90	36.81	22.03

Table 4 Varying Anticipation Time (Seconds) on Epic-Kitchens-55 Accuracy

Model	0.25	0.5	0.75	1	1.25	1.5	1.75	2
RULSTM	39.1	37.3	36.4	35.3	33.4	32.2	30.7	29.5
ImagineRNN	39.1	38.5	36.7	35.6	33.6	32.5	28.8	27.5
Self-Regulated	40.5	38.6	36.7	35.5	34.1	32.3	31.3	30.2
TemporalAgg	41.3	39.5	37.2	36.4	35.1	33.7	31.8	30.9
AVT+	42.3	40.2	37.6	36.8	35.7	34.4	31.9	31.2
LatentGoal	34.3	31.8	27.8	26.9	24.8	24.3	23.0	21.7

Critical analysis

This section will cover general findings from the experimentation. Additionally, each model will be analysed further, relating results to architecture

Table 5 Varying Anticipation Time (Seconds) on EGTEA-Gaze+ (Top-5 Accuracy)

Model	0.25	0.5	0.75	1	1.25	1.5	1.75	2
RULSTM	74.3	71.8	68.4	66.4	63.5	61.4	59.1	56.8
ImagineRNN	74.6	72.3	68.5	66.7	63.4	62.1	59.4	57.2
Self-Regulated	82.6	78.1	73.5	70.7	66.4	64.9	61.8	59.6

Fig.5 shows an aggregation of all model accuracy for each prediction class, split between seen and unseen and Top-1 and Top-5, respectively. Within Fig.5, it is clear that implementation and performance for all classes (Action, Verb and Noun) are higher on the seen test set than on the unseen test set. Additionally, validation and test results yield higher accuracy when using Top-5 as a metric instead of Top-1. Now to address trends within each prediction class (Verb, Noun and Action). As previously described, action labels for this task use compound statements such as ‘get fish’ and ‘cut paper:baking’. With 2,512 unique action representations, the lowest class performance is expected to be the action class. As can be seen, this is consistent across all implementations, dataset training visibility and metrics. Nouns (objects within the scene such as ‘kettle’ and ‘pan’) and Verbs (actions such as ‘open’ and ‘shake’) have consistent rankings of Verb predictions yielding the highest accuracy, with Noun accuracy being the second most accurate class prediction. Just like for action, the rankings for verb and noun are consistent between all implementations, dataset visibility and metrics.

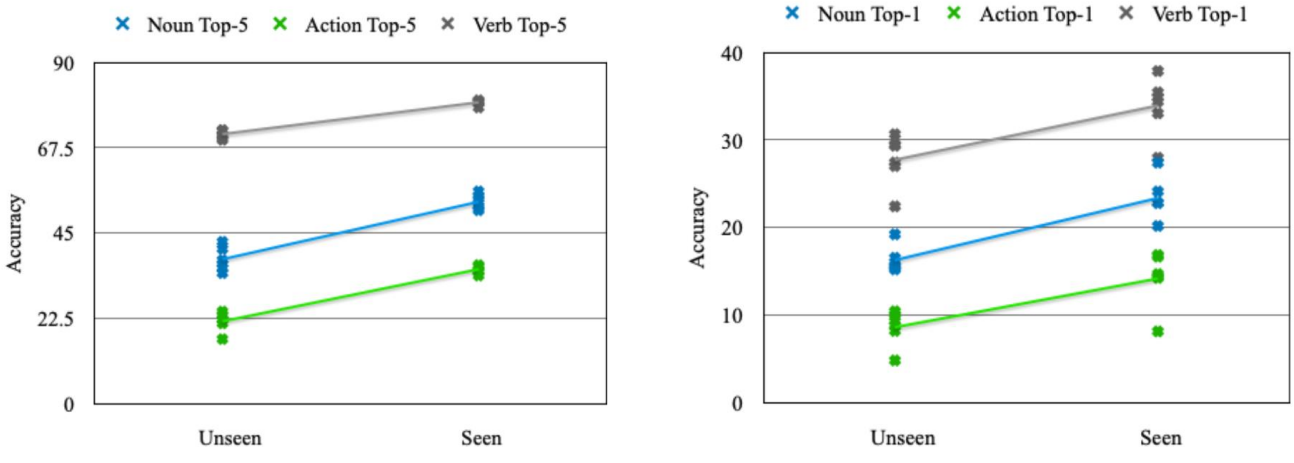

Figure 2 All models scores aggregated into Seen and Unseen

From the results we can infer that, generally, within Epic-Kitchens 55, anticipating the next active object in the scene (anticipating the next object in use) is more complex than anticipating and labelling the following verb. The lack of performance in this class may be due to the poor quality of the object detection or the need for a specialist sub-network to improve the next active object prediction. To summarise the Top1 and Top-5 accuracy in the test set results, the model that achieved the highest performance was the adaptive video transformer (AVT). Latent Goal Aggregation performed second best out of both models in the test set, followed by Temporal Aggregation, which only outperformed all models within Top-1 accuracy Verb prediction within the Seen test-set.

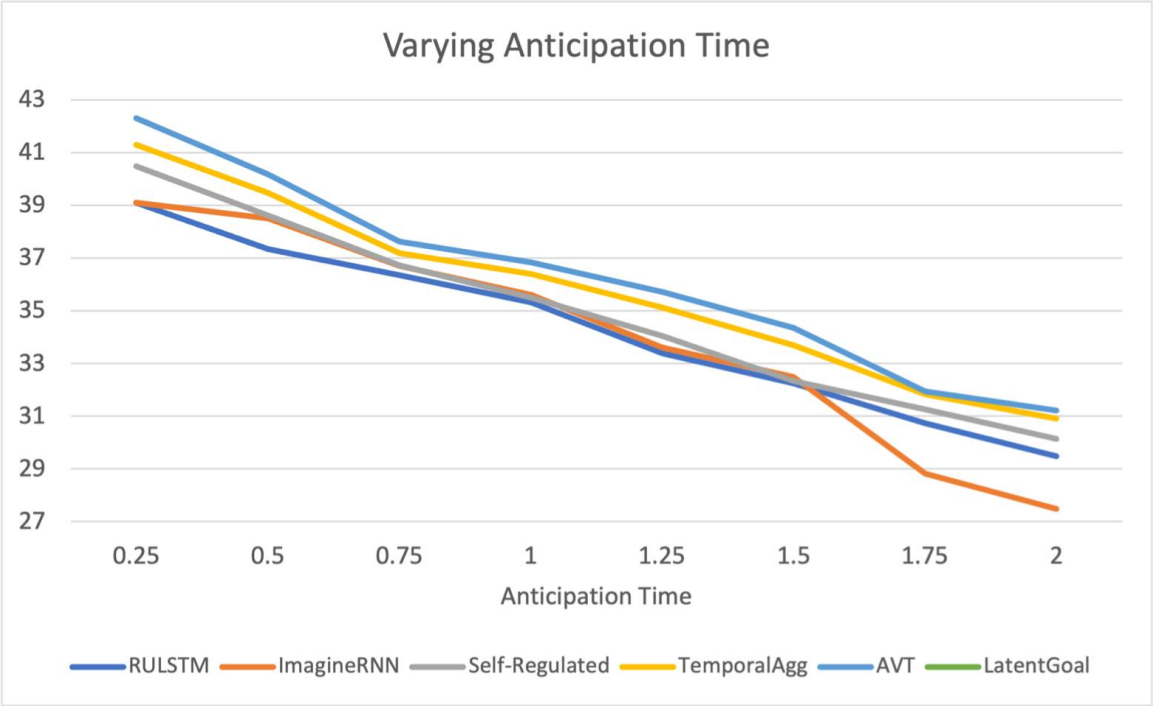

Figure 3 All models tested upon Epic-Kitchens Validation-set, measured with Top-5 Action Accuracy.

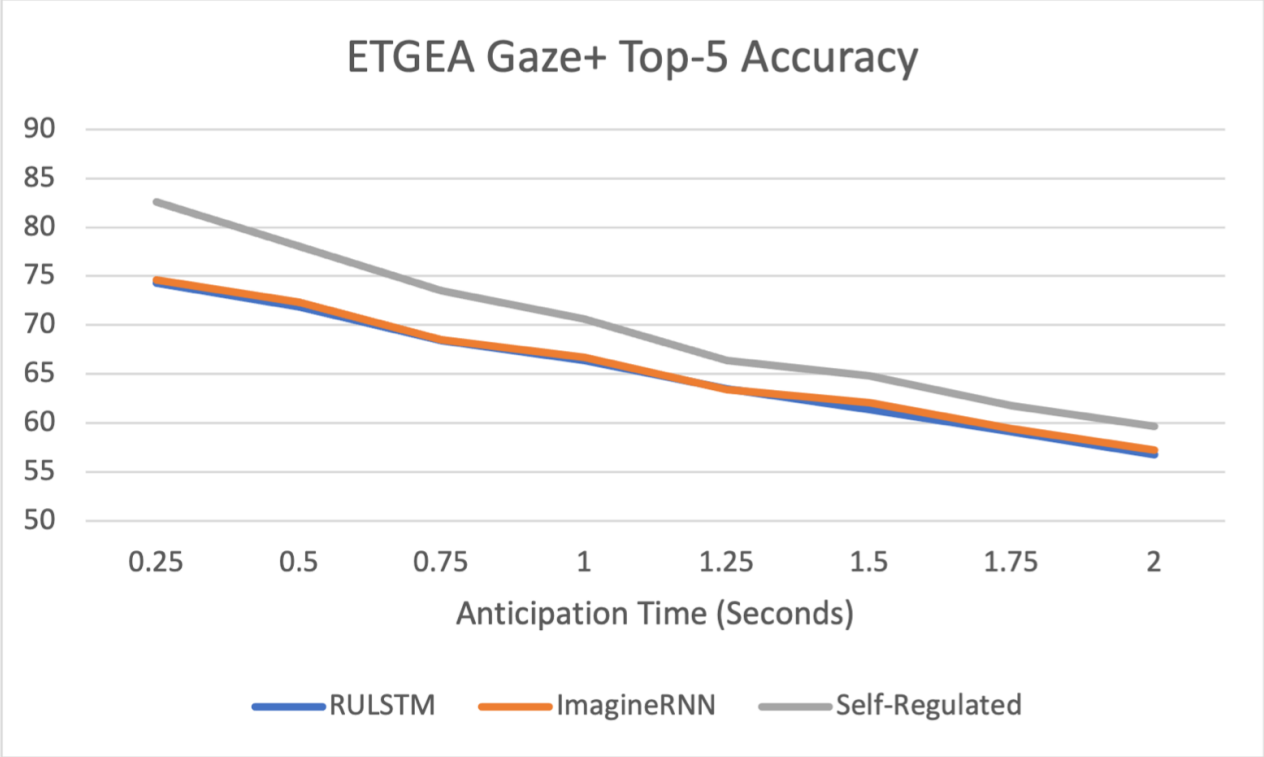

Figure. 4 Iterating anticipation times for EGTEA Gaze+

Fig.3 displays all the models tested on the Epic-Kitchens 55 validation set, with anticipation time iterated (each model is trained to anticipate at 1 second ahead). As expected, the general trend for all models is that action anticipation accuracy drops the greater the time to anticipate. AVT+ performed with the highest accuracy in this task and demonstrated the generalisability of the network. Following with second best results: Temporal Aggregation. This network clearly shows a dominance that is hard to discern from comparing results within the Epic- Kitchens 55 test sets. Self-Regulated Learning, ImagineRNN, and RULSTM have tightly grouped results due to their common recursive sequence prediction architecture. Latent Goal performed the worst. Fig.4 confirms that of the recursive architectures, Self-regulated was the most generalisable when iterating anticipation times compared to RULSTM and ImagineRNN.

RULSTM

The Rolling-Unrolling LSTM (RULSTM) [1] is considered the baseline for this task. The RULSTM has specific tools to improve performance - mainly the 'Sequence Completion Pre- Training' (SCP) and 'Modality Attention mechanism' (MATT). SCP is used to differentiate task specialisation during training. MATT addresses that in specific prediction scenarios, one modality may be more valuable than the other. It does this by calculating attention scores that rate the importance of each modality for each prediction. The attention mechanism does assist the model in filtering uncorrelated features. However, the recurrent nature of LSTMs is limited by the difficulty of training due to long gradient paths [2].

ImagineRNN

ImagineRNN is built using the architectural framework of the RULSTM. Throughout all results, marginal improvements of no more than 1-2% are made over

the RULSTM. While several optimisations are made on top of the RULSTM, the main architectural changes include changing a regression loss function to training the ImagineRNN to pick out the correct future state from distractors, allowing for the learning of how future features change. Additionally, a focus on predicting the difference between adjacent frames is used to lead the model's focus towards features changing between time steps [12]. The improvements by architecture's optimisations are validated within 'Learning to Anticipate Egocentric Actions by Imagination' (Yu et al. [12]). Fig.4 shows RULSTM and ImagineRNN with tightly grouped results indicating that the optimisations of the RULSTM within ImageinRNN does not provide much generalisability. A further limitation of this implementation is that the handling of future uncertainty is absent. A future implementation with the inclusion of the Verb-Noun Marginal Cross Entropy Loss (VNMCE) [19] may address these issues and improve overall model performance.

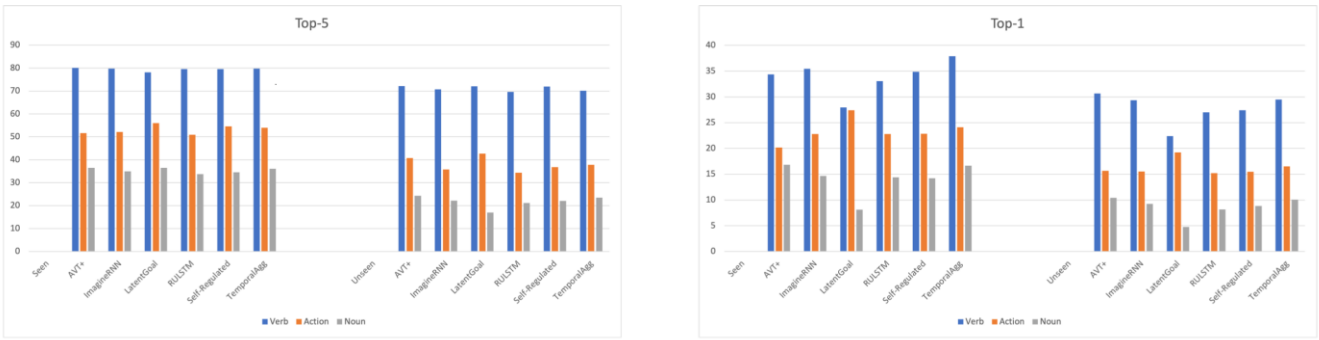

Figure. 5 All results from the Epic-Kitchens Test Set

Self-regulated learning

Self-Regulated Learning did not achieve any best-in-class scores. It outperformed the baseline RULSTM within all classes apart from Top-1 Seen Action, which scored 0.17% behind RULSTM. Compared to the other implementations, its highest ranking was second place within Top-5 Seen Noun (behind Latent-Goal). The performance of Self-Regulated Learning is very similar to ImagineRNN, except having generally lower scores when measured with Top-1 accuracy. While Self-Regulated is architecturally different from the ImagineRNN and, by extension, RULSTM, there are similarities of a recurrent prediction structure (in this instance a layered GRUs) along with optimisations of producing an attention score for each modality and fusing the modalities based on the anticipation results (RULSTM inspired both the attention and the fusion implementations). Self-Regulated Learning can outperform RULSTM and ImagineRNN when varying anticipation times in both datasets, which suggests that the layered GRU recursive structure allows for a more generalisable predictive model than the LSTM implementations. This may result from the multi-task learning framework, which enhances the final video representation by targeting verb, noun, and the final action class.

Latent goal

Out of the models, Latent-Goal is universally the best at noun-related prediction. This may be due to the large number of objects (325) within the dataset compared to a comparatively small 125 verbs. Since Latent Goal aims to model the underlying goal within the sequence to model the following action, a more significant instance of objects (nouns) means that the set of representative goals found while processing the dataset may be mapped more directly onto associative objects than nouns. Further experimentation into the noun performance using a dataset with more

actions than nouns will help determine whether latent goals map better to objects for a fundamental reason or whether more instances allow for clear classification boundaries. The efficiency of this implementation is surprising as the ideal dataset for this model should have as few goals as possible, but due to the natural filming of Epic-Kitchens, the wearer of the first-person view is likely to be pursuing multiple goals simultaneously, for example preparing vegetables while boiling pasta and cleaning up. This parallel goal complexity may explain why the predictions for the action are so comparatively low against all models, falling behind the baseline of RULSTM across all action results. If counting based on best-in-class accuracy, Latent-Goal is considered second place; this is solely based on high noun accuracy and should not be credited as a suitable option for general egocentric action anticipation due to poor overall action anticipation. An exciting application of this implementation may be to incorporate a lightweight version of this architecture as a module on the AVT to solely predict nouns (which, based on this survey, is a weakness in the AVT).

Temporal aggregation

Temporal Aggregation [15] is great at Verb predictions on seen datasets, scoring highest with the Verb class metric within the seen tests set when measured in Top 1% accuracy (Table 4). Measuring within Top-5 accuracy, temporal aggregation's verb class accuracy was second most accurate of all models, 0.26% behind AVT's accuracy. Additionally, it came second within Seen Top-5 Actions and Unseen Actions. The main application for Temporal Aggregation is within third-person action anticipation, specifical anticipation within Breakfast and 50Salads. Breakfast and Salads have a mean clip duration of 26.6 and 29.7 seconds much larger than Epic-Kitchens 55's 3.7 seconds. Furthermore, the number of classes within Breakfast and 50Salads are 48 and 17, respectively, with Epic-Kitchens containing a comparatively large 2513 classes. To effectively infer distant temporal relationships along with such an extensive range of classes shows how adaptive and generalisable this architecture is over a wide range of datasets. As mentioned in Section 4, data representation within the model is fundamentally different. Mixing multiple scales of features from recent and spanning snippets allows the model to anticipate long-range temporal relationships successfully. Within this survey, it performed competitively against the other implementations. This is likely due to the inclusion of verb and noun focus into the models as additional tasks, improving verb accuracy within each testing metric with the most improvement within verb accuracy by 6.5% within Seen Top-1. While attention-based aggregation improves the long-range of temporal dependencies seen within RNN architecture, the model is still limited by creating aggregate representations causing some loss of sequential ordering of sequences.

AVT

AVT [17] produced the best overall anticipation scores, coming best in class in 4/6 categories when measured in Top-5 accuracy and 3/6 categories in Top-1 accuracy (specifically within both Top-5 Seen and Unseen Verb and Action classes). Top1 is best in class within Seen Action and Unseen Verb and Noun. This performance matches the results from the Epic-Kitchens 2021 challenge, where AVT is implemented on Epic-Kitchens 100, a version of Epic-Kitchens 55, but with more footage of different kitchens appended. The AVT also produced the highest results on the Epic-Kitchens 55 validation set for all anticipation times, demonstrating the networks generalisability. This performance over recurrent-based architectures (RULSTM, ImagineRNN, Latent-Goal, Self-Regulated) and the attention-based Temporal Aggregation is unsurprising. The self- attention mechanism for the AVT+ allows for the representation of a sequence to be computed based on relating all features in a sequence to each other, while the multi-headed attention functions to extract information from varying representational

subspace. An example of the efficiency of the attention of the transformer can be seen within an abolition study on the AVT+ [17] which revealed that the model attended explicitly to the hands and objects within the scene 20. Previous works required hand masks as an added modality for the model to focus upon. This focus on hand, objects and their interactions may be critical to the best-in-class verb and action accuracy.

Conclusion and future work

Overall, this work set out to provide a comprehensive survey of egocentric action anticipation models, a niche yet growing area of research. The survey evaluated six different architectures, including three recurrent methods—RULSTM, Self-Regulated, and ImagineRNN. These recurrent models incorporate attention modules to score feature importance and enhance predictions, addressing the challenge that much of the data within a frame is often uncorrelated with the upcoming task. The inclusion of attention mechanisms significantly improves accuracy and generalisability, allowing the models to focus on the most relevant temporal and spatial features. Among the evaluated methods, Temporal Aggregation performed notably well, ranking second overall, due to its ability to effectively aggregate past features using attention-based techniques. However, the current state-of-the-art model, the Anticipative Vision Transformer (AVT), outperformed all other architectures. Built on the transformer architecture, AVT employs a self-attention mechanism, enabling it to process entire sequences in parallel and learn relationships between all features within a sequence. This holistic approach allows AVT to achieve superior accuracy compared to recurrent and other temporal models. In summary, the goals of this study were successfully achieved. This survey provides a comprehensive overview of the current landscape of egocentric action anticipation, highlighting the strengths of self-attention-based models like transformers. By offering insights into model performance and architectural trends, this work serves as a foundation for future research, guiding new implementations towards self-attention mechanisms to further advance the action anticipation task.

References

1.Furnari A, Farinella GM. What would you expect? anticipating egocentric actions with rolling-unrolling lstms and modality attention. In: Proceedings of the International Conference on Computer Vision; 2019. p. 6252–6261.
2.Rodin I, Furnari A, Mavroeidis D, Farinella GM. Predicting the future from first person (egocentric) vision: A survey. Computer Vision and Image Understanding.2021; 211:103252.
3.Furnari A, Farinella GM. Towards streaming egocentric action anticipation. In: 26th International Conference on Pattern Recognition (ICPR). IEEE; 2022. p. 1250–1257.
4.Zatsarynna O, Abu Farha Y, Gall J. Multimodal temporal convolutional network for anticipating actions in egocentric videos. In: Proceedings of the Conference on Computer Vision and Pattern Recognition; 2021. p. 2249–2258.
5.Hochreiter S, Schmidhuber J. Long short-term memory. Neural computation. 1997;9(8):1735–1780.
6.Chung J, Gulcehre C, Cho K, Bengio Y.Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:14123555; 2014.
7.Bulat A, Perez Rua JM, Sudhakaran S, Martinez B, Tzimiropoulos G. Space-time mixing attention for video transformer. Advances in neural information processing systems. 2021; 34:19594–19607.
8.Huang Y, Yang X, Xu C. Multimodal Global Relation Knowledge Distillation for Egocentric Action Anticipation. In: Proceedings of the 29th ACM International Conference on Multimedia; 2021. p. 245–254.
9.Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. Advances in neural information processing systems. 2017;30.
10.Damen D, Doughty H, Farinella GM, Furnari A, Kazakos E, Ma J, et al. Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100. International Journal of Computer Vision. 2022; p. 1–23.
11.Li Y, Liu M, Rehg JM. In the eye of beholder: Joint learning of gaze and actions in first person video. In: Proceedings of the European conference on computer vision; 2018. p. 619–635.
12.Wu Y, Zhu L, Wang X, Yang Y, Wu F.Learning to anticipate egocentric actions by imagination. IEEE Transactions on Image Processing. 2020; 30:1143–1152.
13. Roy D, Fernando B. Action anticipation using latent goal learning. In: Proceedings of the Winter Conference on Applications of Computer Vision; 2022. p. 2745–2753.

14.Qi Z, Wang S, Su C, Su L, Huang Q, Tian Q. Self-regulated learning for egocentric video activity anticipation. IEEE Transactions on Pattern Analysis and Machine Intelligence; 2021.
15.Sener F, Singhania D, Yao A. Temporal aggregate representations for long-range video understanding. In: Proceedings of the European Conference on Computer Vision. Springer; 2020. p. 154–171.
16.Adaloglou N, Karagiannakos S. How Attention works in Deep Learning: understanding the attention mechanism in sequence models. Theaisummer com. 2019;.
17. Girdhar R, Grauman K. Anticipative video transformer. In: Proceedings of the international conference on computer vision; 2021. p. 13505–13515.
18. Damen D, Doughty H, Farinella GM, Fidler S, Furnari A, Kazakos E, et al. Scaling egocentric vision: The epic-kitchens dataset. In: Proceedings of the European conference on computer vision; 2018. p. 720–736.
19.Furnari A, Battiato S, Maria Farinella G. Leveraging uncertainty to rethink loss functions and evaluation measures for egocentric action anticipation. In: Proceedings of the European Conference on Computer Vision Workshops; 2018. p. 389–405.