Universiteit Leiden

Predicting Futures, Not a Future: Diverse Next Action Anticipation

Emmanouil Zagoritis (s4076893)

Supervisors:
Dr. Hazel Doughty & Kaiting Liu

Bachelor Thesis – Data Science and Artificial Intelligence

Leiden Institute of Advanced Computer Science (LIACS)
liacs.leidenuniv.nl

September 2, 2026

Abstract

Action anticipation is the task of predicting a person's next action from an observed video. The same observed context can often be followed by several different but equally plausible actions, but most existing models output only the single most likely action that follows. This thesis investigates whether an existing anticipation model can be modified to produce a set of diverse yet plausible next-action predictions. The baseline model, CLAM, is extended from a single prediction to N parallel prediction slots. Specifically, one anchor slot preserves the primary prediction, while the remaining slots are trained for alternative plausible next actions derived from action-transition statistics and grounded in the objects observed in the video. On EPIC-KITCHENS-100, the final model produces clearly more diverse prediction sets than the baseline while preserving the accuracy of the primary prediction. A human evaluation confirms this diversity gain, but also shows that the additional predictions come at a cost in per-action plausibility.

Contents

1 Introduction

Many systems that interact with people would benefit from the ability to anticipate what those people will do next. In human-robot interaction, a robot that can predict a person's next action results in smoother and more natural collaboration. In autonomous driving, anticipating the behaviour of pedestrians and other drivers can help prevent accidents. Similar ideas are also useful in other domains where systems may need to react before an event actually happens.

Action anticipation is the task of predicting a future action based on an observed video context. It requires a model to reason about the future from incomplete information, which makes the task challenging, as human behaviour is often uncertain. This task can be studied at a *short-term* or *long-term* temporal scale. Short-term action anticipation aims to predict the immediate next action, while long-term action anticipation aims to predict a sequence of next actions over a longer period of time. This thesis focuses on short-term next action anticipation, where the model observes a past context that ranges from a few seconds to roughly one minute, and has an *anticipation gap* of one second, meaning that it predicts the action that starts one second after the last observed frame.

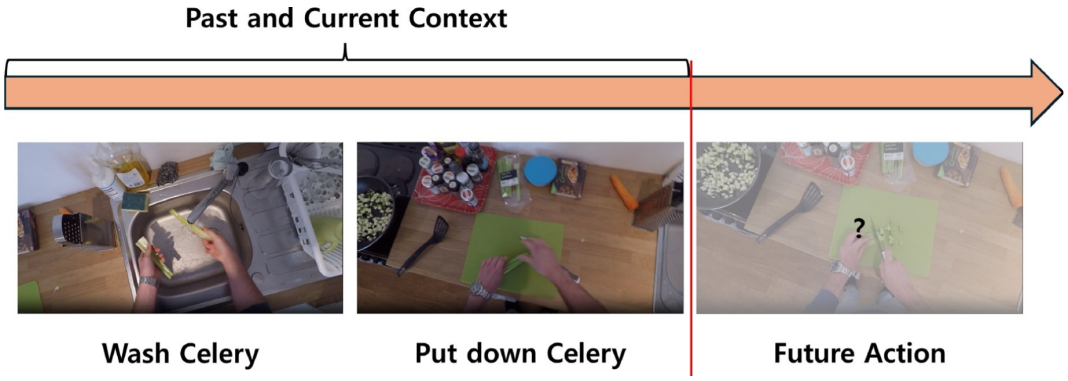

Figure 1: Illustration of short-term action anticipation. Given the observed video context, the goal is to predict the future action before it happens. The same observed context can be followed by multiple plausible next actions (e.g., cut celery, wash carrot, take knife).

There are many situations where the same observed context can lead to several different next actions. For instance, after washing and putting down celery, a person could plausibly continue by cutting the celery, by washing another vegetable, or by reaching for a different utensil. However, most existing action anticipation methods are trained to predict the single action that follows the observed clip, which does not fully represent this uncertainty. As a result, downstream systems are given a single prediction to act on, even when several next actions can be equally reasonable.

Taking the highest-ranked actions from an action anticipation model does not solve this uncertainty either. A model trained to predict one action ranks all actions by how likely each is to be that one action, so its highest-ranked actions tend to be small variations of the same action, for example, different ways of handling the same object. Such a set fails whenever the person turns to a different object, so what is needed is a set that covers genuinely different continuations.

This thesis focuses on diverse next action anticipation. Instead of predicting only one next action, the goal is to output a set of different plausible future actions. The main idea behind this study is that future actions are not always deterministic, and that a useful action anticipation model should represent this uncertainty.

1.1 Research Question

This thesis aims to answer the following research question: *Can a next-action anticipation model be modified to produce diverse yet plausible next action predictions?*

To answer this question, an existing action anticipation model is taken as a baseline and is modified to encourage diverse predictions while constraining them to remain plausible. To support this research question, the following sub-questions will be examined:

- Does a model modified to predict a set of N next actions produce a wider variety of distinct actions than the N highest-ranked predictions of the unmodified model?
- Does such a modified model maintain the accuracy of its primary next-action prediction comparable to the unmodified model, measured with standard action anticipation metrics?
- How does the trade-off between diversity and the accuracy of the predicted set change as the set size N of predicted next actions changes?

1.2 Thesis Overview

This thesis consists of six sections. Section 1 introduces the task, problem definition, and research question; Section 2 provides the related work about the action anticipation task, its recent approaches, and their limitations; Section 3 shows the methodology by describing the baseline model and its modifications; Section 4 describes the dataset used in the thesis, the annotation format, and preprocessing steps; Section 5 presents the experiments and results; Section 6 concludes the thesis and discusses future research. The implementation of all models and experiments is available at https://github.com/zagoritis/CLAM.

This bachelor thesis was conducted at the Leiden Institute of Advanced Computer Science (LIACS) under the supervision of Dr. Hazel Doughty and Kaiting Liu.

2 Related Work

2.1 Single-Prediction Action Anticipation

The action anticipation task has been approached with many different model architectures. Early methods modelled the temporal flow of observed actions with recurrent neural networks, such as LSTMs. An example is the rolling-unrolling LSTM (RULSTM) [FF21], which combines a rolling LSTM to encode the observed sequence and an unrolling LSTM that predicts up to the anticipation point. While this method is effective on short observation windows, recurrent models accumulate errors as they unroll, and they are limited in capturing dependencies between distant frames. To overcome these limits, more recent methods use Transformer-based models, in which self-attention helps relate every observed frame to every other one. The Anticipative Video Transformer (AVT) [GG21] applied this to action anticipation, and since then, Transformers have become the standard backbone. Recent methods build on this foundation by improving different parts of the anticipation pipeline.

Some methods improve how the model attends to the observed video. InAViT [RRF24] focuses on interaction regions, which are the image areas where the hands interact with objects, since these interactions reveal cues about what a person is about to do next. Action-Guided Attention (AGA) [TCP+26] uses the model's own action predictions as queries and keys in the attention mechanism, with past frame embeddings as values, so that history is weighted by what the model expects to come next. The resulting history context is then combined with the current frame embedding through an adaptive gate.

Other methods aim to make Transformer-based anticipation more scalable. The Cross Linear Attentive Memory (CLAM) model [ZMS+26] addresses the cost of self-attention, as it has a computational cost that grows quadratically with the observation length. This model uses a linear-complexity temporal processor with a memory module that maintains a constant memory cost, which allows much longer observation windows.

Another study expands the diversity of training signals seen by the model. ActSeq [QR25] derives a context-free grammar from action sequences in the training set and generates plausible alternative next actions that are not present in the original annotations. Then these annotations are used as extra training samples, which results in reducing the model's bias toward specific action orderings.

All of the methods discussed so far improve the visual representation, the attention mechanism, the scalability, or the training data, but they still target a single next action.

2.2 Uncertainty and Multiple Futures

Other research has focused on the challenge of whether the same observed context can lead to different futures. The methods below do that, from quantifying how uncertain a single prediction is, to generating several possible futures.

One method models the uncertainty in future-action prediction. The Uncertainty-aware Action Decoupling Transformer (UADT) [GAL+24] splits action prediction into a verb-to-noun stream and a noun-to-verb stream that assist each other, quantifies the predictive uncertainty of each, and combines them based on their confidence. This produces a more reliable verb-noun pair prediction without changing the single-prediction output.

The Abstract Goal model [RF24] introduces an abstract goal that captures information about the person's intent, modelled as a learned distribution. Because the next action is also modelled as a distribution, several candidate next actions are sampled for the same observed video, and a goal-consistency mechanism selects the one that matches best with the inferred goal.

Another method frames anticipation as a generative problem. DIFFANT [ZWM+23] applies a diffusion model to action anticipation, where future actions are iteratively denoised from noise, conditioned on the observed video. Because this process is probabilistic, different runs of the same model produce different but plausible future sequences. PlausiVL [MALL24], another generative approach, uses a large video-language model to produce future action sequences, with auxiliary losses that discourage temporally implausible orderings and excessive repetition. Both models are used for long-term sequence generation instead of short-term next-action anticipation.

2.3 Limitations

Three limitations are relevant to the goal of this thesis. Firstly, the most commonly used evaluation metric does not directly measure diversity. The class-mean top-5 recall is a standard metric for action anticipation which rewards a model whenever the ground-truth action is in the top five predictions. As a result, a model that returns five near-duplicate variants of the same most-likely action scores just as well as a model that returns five different next action predictions, provided that both include the ground truth. Even methods that output multiple candidates are not pressured by the metric itself to make the candidate predictions diverse.

Secondly, methods that produce multiple candidates do not explicitly optimize for diversity across a set of alternative predictions for the same next action. Uncertainty-aware models such as UADT [GAL+24] quantify confidence but still target a single prediction. Probabilistic approaches like the Abstract Goal [RF24] and DIFFANT [ZWM+23] sample multiple candidates from a learned distribution, but do not return them as a set, as the first only keeps the most goal-consistent next action, while the second's samples vary by chance and not by design. PlausiVL [MALL24] introduces a repetition loss that discourages duplicate actions, but it operates within a generated long-term sequence. ActSeq [QR25] produces alternative plausible next actions, but uses them only as offline data augmentation, since at inference the underlying model still outputs a single prediction.

Lastly, the trade-off between diversity and plausibility is not directly analysed. A model that outputs only its single most confident guess is highly plausible but not diverse, and conversely, a model that outputs random actions is diverse but not plausible. Useful behaviour lies somewhere in between, and quantifying this trade-off is one of the goals of this thesis.

3 Methodology

This section describes the baseline model and the modifications made to it such that it produces a set of N diverse yet plausible next-action predictions instead of a single most-likely action. First, the baseline architecture is introduced, then it is explained how it is extended to output N candidate predictions, together with the mechanism that pushes these candidates apart from each other. After that, the action-transition prior and object-compatibility are demonstrated, which are the constraints that keep the candidates plausible. Finally, these components are combined into the training objective, and the configuration of the final model is stated.

Problem definition. An observed video is represented as a sequence of frame features $X = (x_1, \ldots, x_T)$ with $x_t \in \mathbb{R}^D$, ending at the anticipation point t_a. The next action starts at $t_a + \tau_a$, where $\tau_a = 1$ second is the anticipation gap. Each action $a \in \mathcal{A}$ is a verb-noun pair (v, n) with $v \in \mathcal{V}$ and $n \in \mathcal{N}$. Standard action anticipation learns a model $f(X)$ that predicts the single action $a^\star \in \mathcal{A}$ starting at $t_a + \tau_a$. This thesis uses a modified action anticipation model to output a set of N actions $\mathcal{S} = \{a^{(1)}, \ldots, a^{(N)}\}$, which should be plausible continuations of X and distinct from each other.

3.1 Baseline Model

The baseline is CLAM [ZMS+26], a recent action anticipation model that was chosen for two reasons. First, it achieves strong performance on EPIC-KITCHENS-100 while remaining efficient to train, which matters because every modification in this thesis requires retraining the model. Second, its decoder already produces predictions from learnable queries, which makes it extensible from one prediction to a set of N predictions.

Baseline Architecture. The modifications described in this section do not depend on a specific model, but they require an anticipation model with two properties. The first property is that the observed video is summarised by an encoder into a set of context tokens. The second is that the prediction is produced by a *future predictor*, which is a non-autoregressive Transformer decoder in the style of DETR [CMS+20] that holds a set of learnable query vectors. Through cross-attention, each query gathers information from the context tokens and is transformed into one future embedding. Each future embedding is then mapped by three independent linear classifiers to logits over the action, verb, and noun label spaces described in Section 4. In a model that predicts a single action, such as CLAM, only one query is used, and thus the model outputs one predicted distribution over the next actions. This decoder is what makes it possible to extend the model from one prediction to a set of N predictions, since more queries can be added without changing anything before them.

In CLAM, the encoder is built from two complementary parts, which are used unchanged in this thesis. The first is a *temporal processor*, which is a stack of four Mamba layers [GD24], which is a recent alternative to the Transformer that processes a sequence step by step, like a recurrent network, while keeping a computational cost that grows only linearly with the sequence length. It enriches each frame feature with information from all preceding frames, producing one action-centric token per frame, and a linear classifier on these tokens is trained to recognise the action taking place at each observed time step, which forces the tokens to carry action-relevant information. Only the most recent action-centric tokens are kept, since they summarise the observation up to the anticipation point. The second part is the *memory module*. Because the temporal processor summarises the video sequentially, visual details from early in the observation window can fade from its tokens even when they are useful for predicting the future. The memory module called CLAM revisits the raw frame features and compresses them into a small fixed number of memory tokens using cross-linear attention, a variant of attention whose memory cost stays constant regardless of the observation length. The most recent action-centric tokens and the memory tokens together form the context tokens that the future predictor attends to.

Baseline objective. The baseline is trained using two losses. The *recognition loss* supervises the temporal processor's per-frame classifier on the observed actions and the *anticipation loss* supervises the single future prediction. The verb and noun heads use standard cross-entropy, while the action head uses an *equalised cross-entropy* [TWL+20]. This is needed because most of the $3,806$ action classes are rare, and standard cross-entropy treats every class other than the correct one as a negative example. Therefore, rare classes get pushed down repeatedly by the common ones and end up almost never being predicted, but the equalised version stops this by ignoring most of these negative signals. Specifically, a class counts as rare when it covers less than $\lambda = 1.76 \times 10^{-3}$ of the training segments, which on EPIC-KITCHENS-100 is every class except the hundred most

common ones. For each training sample, every rare class that is not the correct answer is ignored with probability $\gamma = 0.95$. The original loss scored each class on its own, whereas here the classes compete in a single softmax, so a class is ignored by dropping its score before the softmax is applied. Both of these values are taken unchanged from the baseline. The background class at index 0 is excluded from all losses and metrics throughout. This objective is the same for the primary prediction in every modified model, which is how the accuracy of that prediction is preserved.

3.2 Diverse Predictions

This subsection describes how the baseline is extended to produce N predictions, and the mechanisms that make those predictions different from each other.

From one prediction to a set of N predictions. The initial architectural change of this thesis is to use N learnable queries instead of one. Each query is decoded, in the same forward pass, into its own future embedding and its own action, verb, and noun distributions. The i-th query is referred to as *slot* i, and its action distribution is written as:

$$p^{(i)} = \mathrm{softmax}\big(z^{(i)}\big), \qquad i = 1, \ldots, N, \tag{1}$$

where $z^{(i)}$ are the action logits of slot i. The slots are N alternative hypotheses for the same next action. Architecturally, this extension is small as the decoder already supports multiple queries, and the difficulty is making the slots behave as intended, as nothing in the architecture prevents all N queries from converging to the same answer (later referred to as *slot collapse*). The baseline uses $N = 1$ and the diverse model uses $N = 5$, with other values of N examined in the trade-off analysis in Section 5. Figure 2 shows the modified architecture, including the mechanisms introduced in the rest of this section.

Action similarity. Because every action class is a verb-noun pair, two actions can overlap partially. For example, "wash plate" and "wash cup" share a verb, while "wash plate" and "take plate" share a noun. This structure is captured in a similarity matrix S over all action classes:

$$S(a, b) = \begin{cases} 1.0 & \text{if } a = b, \\ 0.5 & \text{if } a \text{ and } b \text{ share a verb or a noun,} \\ 0.0 & \text{otherwise.} \end{cases} \tag{2}$$

A set of predictions in which every pair has low similarity under S is diverse, because it names different actions on different objects, rather than minor variations of one action. The weight 0.5 means that a pair sharing one component is half as similar as an identical pair, treating the verb and the noun of an action as equally important. Section 5 also reports how often predictions are identical and how often they share a verb or a noun, so the two kinds of overlap can be compared without relying on this weight.

Training-free diverse reranking. Before modifying the model during training at all, diversity can already be improved at inference time. The baseline's single output distribution ranks all actions, and its top-N predictions form a prediction set, but nothing stops these predictions from

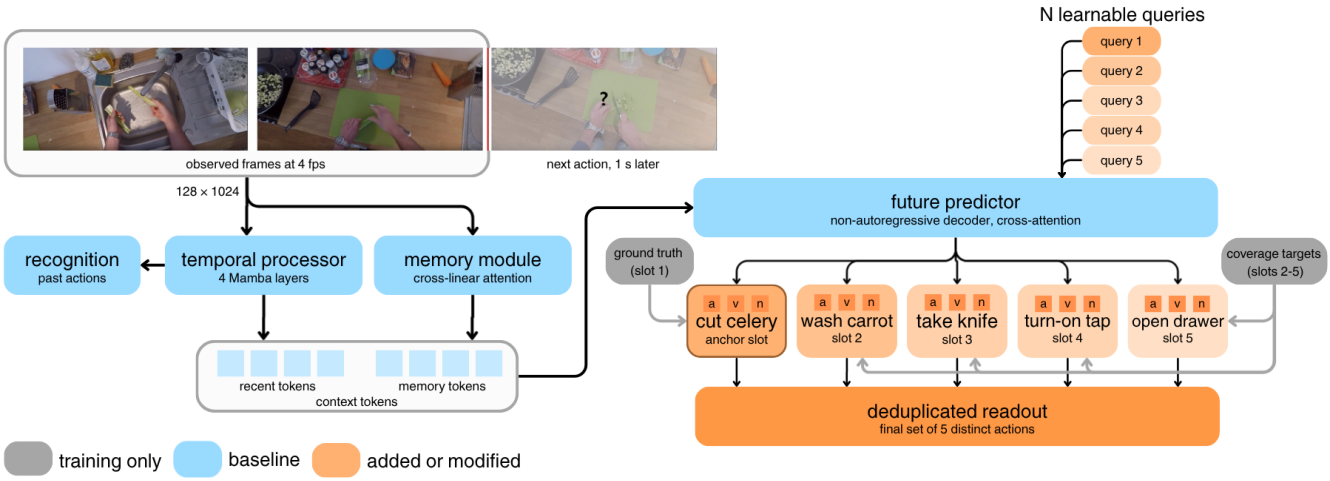

Figure 2: Overview of the modified CLAM architecture, illustrated with $N = 5$ on an observed clip. Precomputed frame features are processed by the Mamba temporal processor, while the memory module revisits them and compresses them into a fixed set of memory tokens. The future predictor decodes N learnable queries into parallel prediction slots, each with action, verb, and noun classifiers (a, v, and n). Slot 1 (anchor slot) is supervised on the ground-truth next action, while slots $2, \ldots, N$ are trained to cover alternative plausible next actions. The grey supervision signals are used only during training, while at inference, the model receives video features alone. Finally, the deduplicated readout converts the N slot predictions into a set of N distinct actions.

being near-duplicates. *Diverse reranking* builds the prediction set greedily, where the first action is the top-scoring one, and each subsequent action is chosen as:

$$a_k \;:=\; \operatorname*{argmax}_{a \notin A_{k-1}} \Big[\, s(a) \;-\; w \max_{b \in A_{k-1}} S(a,b) \,\Big], \tag{3}$$

where $s(a)$ is the model's score for action a, A_{k-1} is the set of actions already selected, and $w \geq 0$ controls how much the similarity to earlier picks is penalised. With $w = 0$, it produces the plain top-N, while larger w adds more variety to the prediction set. This reranking only reorders the outputs of a trained model, so it does not require training, and is used in this thesis both as a simple method for diversity and as the reference against which any trained diversity mechanism is compared.

Hit-anywhere set loss. The first attempt at a set-level objective asks only that some slot predicts the ground truth, implemented as a winner-takes-all loss. For each sample, the slot with the lowest cross-entropy against the ground truth is found, and the loss is $(1 - \varepsilon)$ times the winner's cross-entropy plus ε times the mean cross-entropy over all slots. This exists to protect the non-winning slots from receiving no learning signal at all. The intuition is that unsupervised slots would be free to specialise in alternative futures, but in practice the opposite happens. Because predicting the single most likely action is the best answer for every slot when there is no pressure to differ, all slots collapse onto the same prediction. This objective is retained in the experiments as the demonstration of why an explicit diversity mechanism is necessary, and it motivates the fixed-role design in which every slot always has its own supervision.

Pairwise similarity penalty. The first trained mechanism uses a penalty that discourages the slot distributions from agreeing with each other. For each pair of slots, the expected similarity of their predictions is $\big(p^{(i)}\big)^\top S\, p^{(j)}$, which is high when both slots place their probability mass on the same or related actions. Averaging over all pairs gives the penalty:

$$\mathcal{L}_{\text{sim}} = \frac{2}{N(N-1)} \sum_{i<j} \big(p^{(i)}\big)^\top S\, p^{(j)}, \tag{4}$$

where, for this term only, the slot distributions are computed with a softmax temperature τ, i.e. $p^{(i)} = \text{softmax}\big(z^{(i)}/\tau\big)$. Sharpening the distributions with $\tau < 1$ makes the penalty reflect disagreement between the slots' actual top predictions, rather than being satisfiable by spreading probability mass over the tails. This penalty is intuitive, but it tells the slots to move apart without telling them where to go, which results in unstable training.

Fixed-role coverage. The regulariser is improved with role assignment. Instead of pushing slots away from each other, each slot is given its own target action to cover. Slot 1 acts as the anchor, as it is always supervised on the ground-truth next action, exactly like the baseline's single prediction, so that one high-quality primary prediction is preserved no matter what the other slots do. The remaining slots $2, \ldots, N$ are supervised on a set of $N-1$ coverage targets, which are alternative plausible next actions distinct from each other.

Since the targets form an unordered set, each target must be assigned to exactly one slot before a loss can be computed. This is solved with an optimal one-to-one assignment (Hungarian matching [Kuh55]). Since $N-1$ is small, the implementation enumerates all $(N-1)!$ possible pairings and keeps the one with the lowest total cost. The cost of assigning a target to a slot is the negative log-probability the slot currently gives to that target, and the costs are detached, so the assignment itself carries no gradient. Gradients flow only through the cross-entropy loss of each matched slot-target pair. Let $t_2, \ldots, t_N$ be the coverage targets produced, and let σ be the one-to-one assignment of targets to the non-anchor slots computed by the Hungarian algorithm. The coverage loss is:

$$\mathcal{L}_{\text{cov}} = \frac{1}{N-1} \sum_{i=2}^{N} \text{CE}\big(z^{(\sigma(i))},\, t_i\big), \tag{5}$$

where CE is plain cross-entropy over the action classes, while the equalised variant remains on the anchor only. Since the coverage targets are actions, only the action head of each non-anchor slot is supervised, and the verb and noun heads are trained through the anchor slot. The role assignment step therefore ensures that every slot receives its own target in every sample, rather than having all slots train for the same target. The targets themselves are described in the next subsection.

Deduplicated readout. Even a well-trained set of slots can produce duplicates when two slots' most probable actions coincide on a particular input. For the final inference step, the prediction set is read out sequentially. Slot 1 outputs its most probable action, and each next slot outputs its most probable action among those not already chosen by earlier slots. This guarantees N distinct actions and does not cost anything during training.

3.3 Plausibility Constraints

Diversity itself can be simple because a model that outputs random actions is considered diverse, but it is useless. The mechanisms in this subsection constrain the prediction set to remain plausible, using two types of information. One is the order in which actions tend to follow one another, and the other is the objects visible in the observed video. Both are used as the coverage targets of the fixed-role mechanism.

Action-transition prior. Human activities with a goal are usually sequential, since each action is carried out in the sequence required to reach the desired outcome. For example, in a kitchen, after "turn-on tap", actions such as "wash hands" or "fill kettle" are far more likely than "open fridge". This is captured in a first-order *transition prior*, which is a table $\hat{P}(a' \mid a)$ estimating the probability that action a' follows action a. It is obtained by counting over all consecutive action pairs in the training annotations how often each action follows each other action, and normalising each row into a probability distribution. Each row is smoothed toward the global successor-popularity distribution, so that an action never observed as a predecessor falls back to the overall popularity of next actions rather than a degenerate row. In addition, a small probability floor prevents unseen transitions from receiving exactly zero probabilities inside the logarithm. The transition prior does not depend on the model, as it is computed once before training and never changes. This independence matters because targets derived from the model's own outputs can create a feedback loop in which the model chases a moving target of its own making.

To query the prior for a given video, the previous action is taken to be the last annotated foreground action inside the observation window. The final observed moment can fall in the pause between two actions, where the annotation is background, so the search moves backwards through the window until a foreground action is found. When the whole window is background, the query falls back to the background row of the prior, which has no sequential information. These annotations are used only to construct the training targets, and the baseline already uses the same observed action labels to train its recognition head. At inference, no annotation is used, and the N predictions are produced only from the video features.

Transition-based coverage targets. The transition prior is used during training as the source of the coverage targets in the fixed-role mechanism. For each training sample, the $N-1$ targets for the non-anchor slots are the top-$(N-1)$ most probable continuations of the observed previous action, excluding the ground truth that the anchor slot covers. These targets are external, conditional on the observed context, and distinct by construction, so the extra slots are trained to cover different temporal continuations.

Object-compatibility grounding. A transition that is likely in general can still be impossible in a particular video. For example, "take plate" cannot come next if no plate is anywhere in the scene. The coverage targets are therefore grounded in the observed video. When ranking candidate next actions, a bonus of β is added to the log-probability of every predicted next action whose noun appears among the nouns of the actions observed in the past window:

$$r(a') \;=\; \begin{cases} \log \hat{P}(a' \mid a_{\text{prev}}) + \beta, & \text{if noun}(a') \in \mathcal{O}, \\ \log \hat{P}(a' \mid a_{\text{prev}}), & \text{otherwise,} \end{cases} \tag{6}$$

where $\mathcal{O}$ is the set of observed nouns, and the size of β sets the impact of this constraint. Specifically, a low nats value shows a soft preference, while a value larger than the prior's entire log-probability range (about 28 nats given the probability floor) makes the preference hard, so that any next action compatible to an object outranks every incompatible one. When fewer than $N-1$ compatible next actions exist, the ranking gracefully falls back to the ungrounded prior, such that a full target set is always available.

Noun-distinct target selection. Grounding pulls the targets towards the objects present in the scene, but several likely next actions often involve the same object (for example, "turn-on tap" and "turn-off tap"), which introduces redundancy again. This modification selects the $N-1$ targets greedily under a noun penalty. Candidates are considered in order of the grounded score of Equation 6, and any candidate sharing a noun with an already selected target has its score reduced by a penalty ρ, which is set high to act as a hard constraint. The result is one hypothesis per distinct visible object, and the target set avoids both exact duplicates and partially overlapping ones. When the scene offers fewer distinct nouns than $N-1$, the selection again falls back gracefully by allowing repeats only once distinct options are fully used. At this point, diversity is enforced only over candidates that are identified as more plausible by the transition prior and object grounding. Figure 3 illustrates the full construction for one sample.

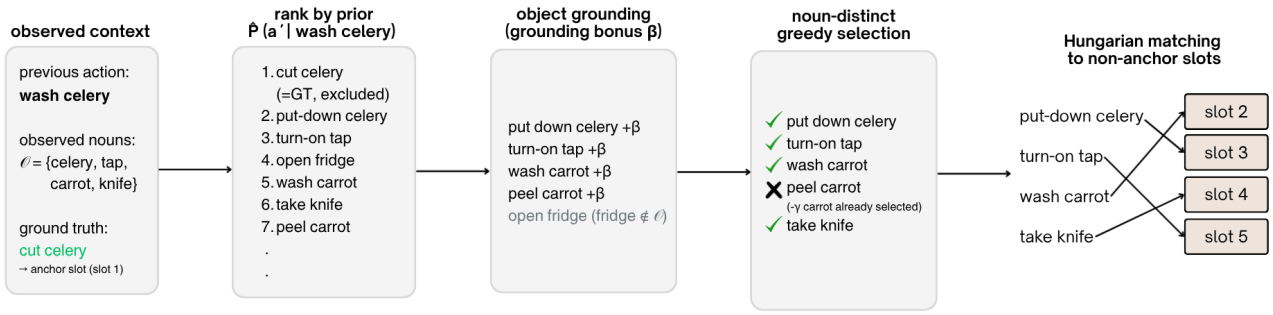

Figure 3: Construction of the coverage targets for one training sample. Candidate successors of the last observed foreground action are ranked by the transition prior, re-ranked by the object-compatibility bonus of Equation 6 so that actions involving observed objects are preferred, and selected greedily under the noun-distinctness penalty so that each target involves a different object. The ground-truth action is excluded, as the anchor slot already covers it. The resulting $N-1$ targets are assigned one-to-one to the non-anchor slots by Hungarian matching.

3.4 Training Objective

The components described above are combined into a single objective. The total objective is:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{anchor}} + w_{cov}(e)\,\mathcal{L}_{\text{cov}}, \tag{7}$$

where $\mathcal{L}_{\text{rec}}$ is the recognition loss, $\mathcal{L}_{\text{anchor}}$ is the baseline anticipation loss applied to slot 1, each summing the action, verb, and noun terms of their respective heads, and $w_{cov}(e)$ is the coverage weight at epoch e. The weight follows a gradual increase, $w_{cov}(e) = w_{cov,\max} \cdot \min(1, e/W)$, with $w_{cov,\max} = 1.0$ and $W = 10$ epochs in the final model, so that the anchor and the underlying representation stabilise before the extra slots are pulled towards their targets. Gradient clipping at 1.0 is also applied throughout for stability.

Final model configuration. The final model uses $N = 5$ slots, transition-based coverage targets with high object grounding ($\beta = 30$), high noun-distinct penalty ($\rho = 100$), the coverage objective of Equation 7, and the deduplicated readout at inference. The training runs for 50 epochs at batch size 128 with the AdamW optimiser without weight decay, using a cosine learning-rate schedule that warms up over the first 5 epochs to a peak learning rate of 0.0003 and decays to a minimum of 10^{-7}, with gradient clipping at 1.0, mixup augmentation, and bfloat16 precision on a single A100 GPU. The peak rate is reduced from the baseline's 0.0005 (explained in Section 5). Checkpoints are selected by validation diverse set recall over the $N = 5$ predictions, while the baseline is selected by class-mean top-5 recall. The training-free reranking (Equation 3) is not part of the final model, but it is evaluated as a reference for the trained model.

4 Dataset

The experiments in this thesis use EPIC-KITCHENS-100 [DDF+22], a large-scale egocentric dataset of unscripted daily kitchen activities. It contains 100 hours of video recorded by 37 participants across 45 kitchens using head-mounted cameras and provides about 90K action segments annotated in 700 long untrimmed videos. A few examples from this dataset are shown in Figure 4.

Figure 4: Example frames from EPIC-KITCHENS-100 [DDF+22] with their verb-noun action labels. Each frame illustrates a head-mounted viewpoint and the hand-object interactions.

The EPIC-KITCHENS-100 dataset suits well to the goal of this thesis for two reasons. First, it is one of the standard benchmarks for short-term action anticipation and is used by the baseline model, which keeps the results of this thesis comparable to prior work. Second, for diverse action anticipation, its activities are unscripted and densely annotated, so the same observed context could be followed by several different but equally reasonable actions.

4.1 Annotations and Splits

Each annotation in EPIC-KITCHENS-100 describes a single action instance. Specifically, it records which participant and video it comes from, the original free-form narration, the start and end of the action given as both timestamps and frame indices, and the verb and noun parsed from the narration together with their class identifiers. The action class is derived from the verb-noun pair, drawn from 97 verb and 300 noun classes, of which $3,806$ verb–noun combinations occur in the dataset.

Additionally, the dataset is divided into training, validation, and test splits, with $67,217$ annotated action segments across 495 videos for training and $9,668$ segments across 138 videos for validation. This thesis trains on the training split and evaluates on the validation split, since the test split labels are not public.

All classes without filtering are used in this thesis. The long-tail class imbalance is instead handled by the training objective. A background class at index 0 is added to each label space and treated as an ignore index, masked out of both the loss and the evaluation metrics.

For the anticipation task, each annotated action becomes one sample, where the model sees a fixed window of a video that ends one second before the action starts, and has to predict that action based on that observed window only, without ever observing the action itself.

4.2 Data Preprocessing

Raw video is not input to the model directly, but instead each frame is represented by a precomputed feature vector, following the standard pipeline established by RULSTM [FF21]. Frames are encoded with the TSN BN-Inception RGB features released by RULSTM [FF21], trained for action recognition on EPIC-Kitchens. These features are not that recent, but they remain the standard input for this benchmark and are also used by the baseline [ZMS+26], so any differences measured come from the model and not the visual backbone. The encoder is frozen, so features are extracted once and stored on disk, instead of being recomputed during training, and each frame yields a 1024-dimensional feature vector.

Features are sampled at 4 fps, meaning one feature vector every 0.25 seconds. Each feature is computed from a single RGB frame [FF21], so all temporal information comes from the sequence. For each action to be anticipated, the observation window ends one second before the action starts (anticipation gap $\tau_a = 1$s) and spans the preceding $\tau_o = 32$ seconds, composed of 31 seconds of long-term context and a final 1 second of working memory. At 4 fps, this corresponds to a fixed sequence of $T = 128$ feature vectors per sample.

4.3 Input & Output Format

Each sample is presented to the model as a single tensor of frame features of shape $[B, T, D]$ with $T = 128$ and $D = 1024$, stored in single precision. The T dimension is the ordered sequence of 4 fps RGB features, and the D dimension is the BN-Inception feature vector of each frame. Positional information is added inside the model, and the most recent frames are distinguished from the longer-term context internally by the baseline model's architecture. The features are passed to the model unchanged, and the input is used directly by the encoder.

To ensure a fixed input shape, observation windows are padded to length $T = 128$ by repeating the earliest available frame when fewer than 128 frames precede the anticipation point, and are cropped to the most recent frames when more are available. Samples whose action starts less than the anticipation gap into the video, leaving no observable past, are removed during the conversion to anticipation windows.

For each input, the model produces predictions through three independent linear classifiers. One per label space is applied to each of the N predicted future embeddings produced by the decoder. The three output tensors have shapes $[B, N, 3807]$ for actions, $[B, N, 98]$ for verbs, and $[B, N, 301]$ for nouns (each class dimension is one larger than the corresponding number of foreground classes, because index 0 is reserved for the background class).

N is the number of next-action predictions the model produces for each sample, and each prediction comes from a learnable decoder query. In the baseline model, $N = 1$, so only one next action is predicted, while in the diverse-set model, $N = 5$, making the model generate five candidate next actions.

5 Experiments

This section evaluates the mechanisms described in Section 3 on the EPIC-KITCHENS-100 validation set described in Section 4. First, the evaluation metrics are defined, and after that, the quantitative results are presented in the same order in which the mechanisms were introduced. Then, the diversity-accuracy trade-off is analysed as a function of the set size N, an ablation study isolates the contribution of each component, and lastly, there is a human evaluation that rates the predictions of both models.

5.1 Evaluation Metrics

The research question has two parts, plausibility and diversity, and each part has its own metrics. Diversity is measured directly on the prediction set, while plausibility is approximated by standard accuracy metrics against the single annotated next action and assessed directly in the human evaluation. Another type of diagnostic metric is used to detect specific failure modes that appeared during the development.

For a validation set of B samples, y_b is the annotated next action of sample b and $\hat{A}_b = \{\hat{a}_b^{(1)}, \ldots, \hat{a}_b^{(N)}\}$ is its prediction set, with $\hat{a}_b^{(i)}$ coming from slot i. All metrics below are averaged over samples, except the class-mean top-5 recall, which is averaged over action classes.

Accuracy metrics. Accuracy is measured with the standard action anticipation metrics of the EPIC-KITCHENS-100 benchmark, which makes the results comparable to prior work. These metrics compare the predictions against the single annotated next action, so they are used as an approximation of plausibility and not a direct measure of it.

Class-mean top-5 recall (MT5R) is the main evaluation metric used in EPIC-KITCHENS-100 [DDF+22], following the definition of RULSTM [FF21]. For each action class c, top-5 recall measures the percentage of samples where the correct class c appears among the five predicted actions. Class-mean top-5 recall is then calculated by averaging this value across all action classes. The top-5 metric is useful because future actions are uncertain, and several different actions may be possible given the same observed context. Averaging across classes also prevents the frequent classes from dominating the score, and EPIC-KITCHENS-100 includes all action classes in this average, rather than only the most frequent classes. Previous work reports MT5R separately for verbs, nouns, and actions, and this thesis reports the action score. For multi-slot models, MT5R is calculated using only the anchor slot, allowing a direct comparison of the main prediction with the baseline's single prediction.

Top-N set recall (top-N) measures how often the ground truth appears among the top N actions of the anchor slot's distribution. This is the natural score of the prediction set that a single-head model produces by simply taking its N highest-ranked actions.

Diverse set recall (diverse@N) is not a standard action anticipation metric and is introduced in this thesis because the modified model outputs one action per slot instead of a ranked list. It measures how often the ground truth appears among the N actions that the model actually outputs as its prediction set, one action per slot:

$$\text{diverse@}N = \frac{1}{B}\sum_{b=1}^{B} h_b, \qquad h_b = \begin{cases} 1 & \text{if } y_b \in \hat{A}_b, \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

Top-N set recall uses the same formula, with $\hat{A}_b$ being the N highest-ranked actions. For the baseline ($N = 1$ extended to a top-N list), diverse set recall and top-N set recall are the same. For the multi-slot models, these are different, and their difference shows how much recall is lost by adding diversity to the predictions.

Diversity metrics. Diversity is measured on the prediction set itself, using the action similarity matrix S of Equation 2.

Set similarity (SetSim) is the average pairwise similarity under S over all $N(N-1)/2$ pairs of actions in the prediction set:

$$\text{SetSim} = \frac{1}{B}\sum_{b=1}^{B} \frac{2}{N(N-1)} \sum_{i<j} S\left(\hat{a}_b^{(i)}, \hat{a}_b^{(j)}\right). \tag{9}$$

When its value is 0, it means that every pair of predictions involves a different verb and a different noun, and a value of 1 means that all predictions are identical.

Duplicate rates split the set similarity into its causes. Exact duplicates (*ExactDup*) are the fraction of predictions in the set that are duplicates of another prediction:

$$\text{ExactDup} = \frac{1}{B}\sum_{b=1}^{B} \left(1 - \frac{\left|\text{unique}(\hat{A}_b)\right|}{N}\right), \tag{10}$$

Verb duplicates (*VerbDup*) and noun duplicates (*NounDup*) are the fractions of pairs that share a verb or a noun:

$$\text{VerbDup} = \frac{1}{B}\sum_{b=1}^{B}\frac{2\,d_b^{\text{verb}}}{N(N-1)}, \qquad \text{NounDup} = \frac{1}{B}\sum_{b=1}^{B}\frac{2\,d_b^{\text{noun}}}{N(N-1)}, \tag{11}$$

where d_b^{verb} and d_b^{noun} are the numbers of prediction pairs in $\hat{A}_b$ that share a verb and that share a noun. These support the set similarity metric, as a set can avoid exact duplicates but repeat the same object with different verbs, and these rates can spot that. The set similarity is determined by these two rates, as $\text{SetSim} = (\text{VerbDup} + \text{NounDup})/2$.

Diagnostic metrics. Diagnostic metrics detect failures that diversity metrics cannot spot.

Object match (ObjMatch) is the percentage of predicted next actions whose noun appears among the set $\mathcal{O}_b$ of nouns annotated anywhere in the observation window, which is the same test used for the grounding bonus of Equation 6:

$$\text{ObjMatch} = \frac{1}{B}\sum_{b=1}^{B}\frac{o_b}{m_b}, \tag{12}$$

where o_b is the number of predictions whose noun appears in $\mathcal{O}_b$, and m_b is the number of predictions with a defined noun. This metric measures whether the predictions are grounded in the objects that are actually present in the clip.

Cross-clip overlap (CrossClip) is the average fraction of one sample's non-anchor predictions that also appear among another sample's non-anchor predictions. It is computed inside each evaluation batch of M samples, over all ordered pairs $i \neq j$:

$$\text{CrossClip} = \frac{1}{M(M-1)}\sum_{i\neq j}\frac{c_{ij}}{N-1}, \tag{13}$$

where c_{ij} is the number of sample i's non-anchor predictions that also appear among sample j's non-anchor predictions. The anchor slot is excluded on both sides, since it is the primary prediction and is expected to repeat across clips. The reported value is the average of these per-batch values over the validation set, so it depends on the batch size, which is 128. If the model outputs input-dependent hypotheses, this value should be close to 0, and if the extra slots predict the same popular next actions for every video, it is close to 1.

5.2 Quantitative Results

Baseline and training-free reranking. The retrained CLAM baseline reaches an MT5R of 16.39 at the final epoch, and its top-5 prediction set achieves a set recall of 24.88. This top-5 set is already quite diverse, with set similarity 0.206, and it contains no exact duplicates, since it has no duplicates by construction. However, it contains partial repeats, with about 19% of pairs sharing a verb and 22% sharing a noun. This baseline top-5 set is the reference against which every modification is compared.

The diverse reranking of Equation 3 improves this set without any retraining. Table 1 shows the effect of increasing the reranking weight w.

	SetSim	VerbDup	NounDup	diverse@5	MT5R
Baseline top-5	0.206	19.37	21.82	24.88	16.39
Reranking ($w = 0.5$)	0.182	16.68	19.67	24.78	16.39
Reranking ($w = 1.0$)	0.156	13.95	17.27	24.53	16.39

Table 1: Diverse reranking of the baseline's output distribution for increasing the reranking weight w. Diversity improves a lot with a recall cost of 0.35 points and no change to the model. MT5R is identical for all, as reranking only reorders the final epoch predictions of the same trained checkpoint. Table 5 sweeps w further and compares it against the final model.

At $w = 1.0$, set similarity drops by 0.05 and both duplicate rates fall, at a recall cost of 0.35 points. Since reranking trades recall for diversity without any retraining, a trained diversity mechanism is only worth its cost if it reaches a given set similarity at a smaller recall cost than reranking needs to reach that same set similarity. The next subsection sweeps w and makes this comparison.

Modifications to the final model. The trained model configurations are reported in two tables. Table 2 summarises the set-level objectives, and Table 3 covers the coverage-target mechanisms that lead to the final model. Each row corresponds to one mechanism from Section 3.

The *hit-anywhere set loss* confirms the failure described in Section 3. In this experiment $\varepsilon = 0$, so only the best-matching slot gets trained for each sample. Although it only requires that some slot predicts the ground truth, all five slots collapse to the same next action prediction. Specifically, the set similarity reaches 0.990, meaning that the five slots are almost always identical, and the diverse set recall is 8.59. This collapse demonstrates the need for a diversity mechanism.

The *pairwise similarity penalty* of Equation 4 does not fix this collapse. At a moderate weight, the slots remain collapsed, and increasing the weight and decreasing the temperature makes training unstable instead of diverse. The set similarity is between near 0 and near 1 across training, and both recall and MT5R decrease. The problem here is that the penalty tells the slots to move apart from each other without telling them where to go.

Model	diverse@5	top-5	SetSim	MT5R
Baseline top-5 (reference)	24.88	24.88	0.206	16.39
Hit-anywhere set loss	8.59	23.16	0.990	15.77
Pairwise similarity penalty (w_{sim}=0.5)	8.75	23.41	0.989	16.20
Pairwise similarity penalty (w_{sim}=2.0, τ=0.5)	5.18	16.38	0.981	11.86

Table 2: Set-level objectives at $N = 5$. All three trained configurations collapse the slots onto the same prediction. All these models were trained on a single A100 GPU and are reported at their final epoch with the raw argmax readout.

The *fixed-role coverage* mechanism solves the collapse problem. With self-derived targets (each non-anchor slot matched to one of the anchor's own top non-ground-truth predictions), set similarity drops to 0.415, and diverse set recall reaches 15.88. However, this method has a structural ceiling

because the extra slots imitate the anchor slot's predictions. Thus, the set can only approach the anchor slot's top-5 but never exceed it, and it remains less diverse than the baseline's top-5 prediction set.

Next, the self-derived targets are replaced with the *transition-based coverage targets*, which break this ceiling. Diverse set recall rises to 19.37, and set similarity drops to 0.124. Self-derived targets can only teach the extra slots to reproduce what the anchor slot already knows, whereas the transition prior provides information that the visual model does not have. This is the most diverse trained set, with the anchor slot intact, and achieves an MT5R of 16.53. Ranking the next actions by $\hat{P}(a' \mid a_{\text{prev}})$ with the ground-truth previous action recalls the true next action 34.9% of the time in its top 5. This is an oracle diagnostic, as it uses the annotated previous action that the model does not have at inference. This is higher than the visual model's own set recall, which shows that the ordering of kitchen actions carries information that the video features alone do not provide, making the prior a good source of coverage targets.

Unfortunately, some qualitative analysis showed that most of this diversity was not really input-dependent. This failure and its fix are analysed later in the ablation study. Adding the grounding mechanism achieves input-dependent next action predictions, with cross-clip overlap dropping from 0.6-0.8 to 0.167, and the set similarity 0.207 close to the baseline's, because the grounding concentrates the predicted nouns on the objects present in the scene. Since a clip contains only a few visible objects, the slots start repeating nouns and differ in the verb instead, which is what the next mechanism fixes.

Finally, the *noun-distinct target selection* pushes the next action predictions apart from each other while keeping the grounding. With this mechanism, noun duplicates fall from 29.8% to 8.9% of pairs, and set similarity drops to 0.142, making this the first trained model whose prediction set is more diverse than the baseline top-5 (0.206). In addition, its diverse set recall of 22.13 is almost unchanged from the grounded model's 22.49. Therefore, the gain in diversity comes at almost no cost in recall.

Model	diverse@5	top-5	SetSim	MT5R
Baseline top-5 (reference)	24.88	24.88	0.206	16.39
Fixed-role coverage (self-derived targets)	15.88	22.44	0.415	15.80
Fixed-role coverage (transition targets)	19.37	24.59	0.124	16.53
+ object-compatibility grounding	22.49	24.30	0.207	15.84
+ noun-distinct selection	22.13	24.40	0.142	16.59

Table 3: Coverage-target mechanisms at $N = 5$, in order of development. The last row is the final model. All these models were trained on a single A100 GPU and are reported at their final epoch with the raw argmax readout.

Answering the sub-questions. Table 4 compares the final model directly against the baseline on the metrics of both sub-questions.

The first sub-question asks whether the modified model produces a wider variety of distinct actions than the N highest-ranked predictions of the unmodified model, and it does. Set similarity is 31% lower, noun duplicates are reduced by a factor of 2.5, and verb duplicates are equal. The remaining exact duplicates (7.8% of predictions) are removed by the deduplicated readout.

The second sub-question asks whether the modified model maintains the accuracy of its primary prediction, which it achieves. The anchor slot's MT5R of 16.59 is comparable to the baseline's 16.39, and the anchor's top-5 set recall of 24.40 is almost the same as the baseline's 24.88. Since the final model trains at a lower learning rate peak than the baseline, an experiment verified that the comparison is fair. The baseline retrained at the lower learning rate performs worse (final epoch MT5R 16.00, against 16.39 at its own learning rate), so both models are compared at their own best setting.

There is one drawback of the modified model compared to the baseline. The recall of the full five-prediction set (22.13) is 2.7 points below the baseline's top-5 recall (24.88). This is expected, as the baseline uses all five predictions on the most likely next action, while the final model uses four of them on alternative predictions that are counted as wrong whenever a different action follows.

	Baseline top-5	Final model
Diversity		
SetSim	0.206	0.142
ExactDup (%)	0.0	7.8
VerbDup (%)	19.4	19.4
NounDup (%)	21.8	8.9
Accuracy		
MT5R	16.39	16.59
top-5 (anchor)	24.88	24.40
diverse@5	24.88	22.13

Table 4: Final model vs. baseline at the final epoch, using the raw argmax readout. The final model produces more varied predictions while keeping the accuracy of its top prediction. The deduplicated readout used at inference is reported in Table 7.

5.3 Diversity-Accuracy Trade-off

The third sub-question asks how the trade-off between diversity and accuracy changes with the different number of next action predictions. The trade-off can be adjusted in two ways, the reranking weight w, which does not require retraining, and the set size N, which does.

Reranking weight. Table 5 sweeps the reranking weight w of the baseline model, so that the trained model can be compared against reranking at the same diversity. Reranking reaches a fully distinct set at $w = 10$, and at $w = 2.0$, its set similarity is 0.106, close to the final model's 0.102, but its diverse set recall is 23.66 against the final model's 21.77. So on the trade-off alone, the trained mechanisms do not perform better than reranking.

Two differences do not show up in this trade-off. First, at the same set similarity, the final model has less than half the noun duplicates (4.9% against 12.3%), so its predictions cover different objects, while reranking spreads its diversity over both verbs and nouns. Second, the object match of reranking drops from 31.4 to 21.8 as w grows, because reordering one distribution cannot add information about the observed scene, while the final model keeps an object match of 31.0. Reranking also leaves the primary prediction unchanged, so its class-mean top-5 recall stays at the baseline, while the final model's anchor slot reaches 16.67.

w	SetSim	VerbDup	NounDup	diverse@5	ObjMatch	MT5R
0 (top-5)	0.206	19.4	21.8	24.88	31.4	15.9
0.5	0.182	16.7	19.7	24.83	31.0	15.9
1.0	0.156	14.0	17.3	24.60	30.5	15.9
1.5	0.130	11.3	14.8	24.13	29.8	15.9
2.0	0.106	9.0	12.3	23.66	29.0	15.9
3.0	0.065	5.2	7.8	22.44	27.1	15.9
5.0	0.019	1.6	2.3	20.47	23.9	15.9
10.0	0.000	0.0	0.0	18.01	21.8	15.9
Final model (argmax)	0.157	20.8	10.6	20.32	32.6	16.68
Final model (dedup)	0.102	15.5	4.9	21.77	31.0	16.67

Table 5: Diverse reranking of the baseline's output distribution for increasing reranking weight w, with the final model shown for comparison. Reranking reaches every set similarity of the final model at a higher diverse set recall. MT5R is unaffected by w, since reranking only reorders one distribution, and is reported as a single value because the small differences across the sweep are run-to-run variation. These evaluations were run locally on a different GPU, so their absolute values are not comparable to Tables 1, 2, 3, 4 and 6, but they are comparable to Table 7.

Different set sizes. The final modified model was retrained from scratch with different set sizes for $N \in \{3, 5, 7, 10\}$. Table 6 reports the converged results.

N	diverse@N	top-N	SetSim	ExactDup	VerbDup	NounDup	ObjMatch	CrossClip	MT5R
3	18.07	19.71	0.165	6.3	16.3	16.8	48.6	0.119	15.83
5	22.13	24.40	0.142	7.8	19.4	8.9	43.4	0.191	16.59
7	24.48	26.74	0.136	7.8	21.8	5.5	37.9	0.239	15.95
10	20.92	30.87	0.371	48.8	43.8	30.5	37.5	0.173	16.76

Table 6: Results of the final modified model at its final epoch with the raw argmax readout, using different set sizes of next action predictions. The trade-off is controllable and effective up to $N = 7$, but breaks at $N = 10$ as it collapses mid-training, leaving duplicates in nearly half of all next action predictions.

Firstly, coverage grows with N up to a point. Diverse set recall rises monotonically from 18.1 at $N = 3$ to 24.5 at $N = 7$, and its difference with the single-head top-N stays consistently at about 2 points.

Second, the anchor slot's MT5R stays similar across all N (15.8-16.8, all around the baseline's 16.39). Adding slots does not hurt the primary prediction at any set size.

Third, within-set diversity also improves with N up to 7 (set similarity drops from 0.165 to 0.136, and noun duplicates drop from 16.8 to 5.5), which is explained by the fact that more slots cover more distinct objects. The $N = 3$ set looks worse on noun duplicates, as two of its three prediction pairs involve the anchor slot prediction, which is never noun-deduplicated. Verb duplicates increase slightly with N (increases from 16 to 22) because more objects end up sharing the few most common kitchen verbs.

Lastly, each extra slot is only slightly worse than the one before it. Specifically, Object grounding weakens with every extra slot (ObjMatch moves from 48.6 to 37.9), since later targets sit deeper in the list of compatible objects, and cross-clip repetition grows (increases from 0.12 to 0.24), because the later new predictions are more generic kitchen actions instead of clip-specific next actions.

For the $N = 10$ set, slot collapse begins around the point at which the learning rate reaches its peak, leaving 48.8% of the predictions as duplicates and a recall difference of around 10 points. Therefore, the trade-off is controllable up to $N = 7$, with $N = 5$ being the best set size overall, as it has the best anchor slot prediction, good object grounding, and a balanced cross-clip repetition, and $N = 7$ being a decent higher-coverage operating point. Figure 5 visualises all the different set size results.

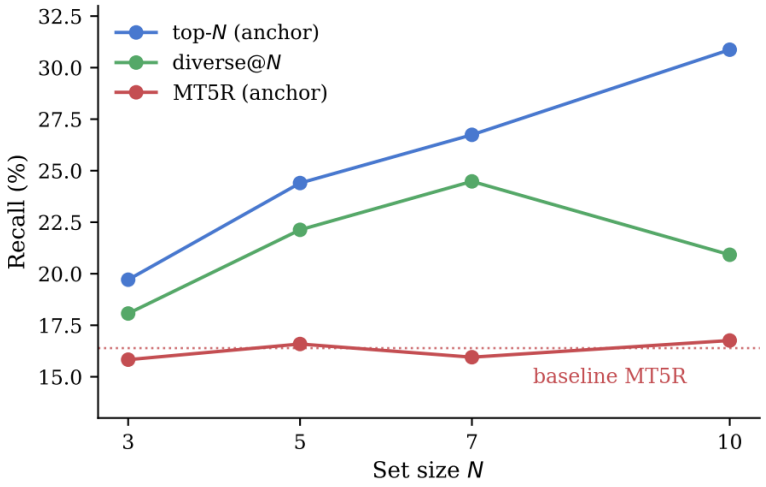

Figure 5: Effect of the set size N using the final model. Diverse set recall increases with N while staying about two points below the anchor slot's top-N, so each extra slot adds coverage at a small constant cost. The anchor slot's MT5R stays at the baseline level (dotted line). At $N = 10$, the training partially collapses, so the trade-off is controllable up to $N = 7$.

Effect of the deduplicated readout. The deduplicated readout is applied in the evaluation of the model and can be switched on for any checkpoint. Table 7 measures its effect per set size on the best checkpoints.

The readout behaves identically at every N. It removes all exact duplicates, decreases other duplicate metrics, raises diverse set recall because a duplicate can never add coverage, and leaves the anchor slot's MT5R unchanged, with an object grounding cost of below 2 points. The benefit grows with N, from +0.8 recall at $N = 3$ to +2.0 at $N = 7$, because larger sets contain more collisions to fix. The final diverse model uses the deduplicated readout, and the human study evaluates the model in this configuration.

N	readout	diverse@N	SetSim	ExactDup	VerbDup	NounDup	ObjMatch	MT5R
3	argmax	17.82	0.145	5.5	13.7	15.4	40.4	15.58
3	dedup	18.62	0.106	0.0	9.4	11.9	39.5	15.58
5	argmax	20.32	0.157	11.0	20.8	10.6	32.6	16.68
5	dedup	21.77	0.102	0.0	15.5	4.9	31.0	16.67
7	argmax	21.58	0.203	17.7	29.4	11.1	26.8	15.55
7	dedup	23.60	0.130	0.0	23.2	2.9	24.9	15.55

Table 7: Effect of the deduplicated readout for each set size, measured on the best checkpoints ($N = 10$ is excluded because of the training collapse). These evaluations were run locally on a different GPU, so their absolute values are not comparable to Tables 1, 2, 3, 4 and 6, but they are comparable to Table 5. Within each pair, the argmax row is the matched control for the dedup row, so only the difference inside a pair should be read. The readout removes all exact duplicates, improves every diversity metric, increases the recall, and leaves the anchor slot unchanged with an object grounding cost of below 2 points.

5.4 Ablation Study

This subsection isolates the contribution of each component of the final diverse model by changing one variable at a time. This model consists of the source of the coverage targets, the object-compatibility grounding, the peak learning rate, the noun-distinct selection, and the readout deduplication.

Coverage-target source. Comparing the two fixed-role rows in Table 3 isolates the effect of the source of the coverage targets, while everything else remains the same.

Object-compatibility grounding. The transition-coverage model without grounding has a failure that the diversity metrics cannot see, which motivated both the grounding and the cross-clip overlap metric. After doing a qualitative analysis of its next action predictions, it was shown that the non-anchor slots produce the same popular next actions for almost every video. Specifically, on the 50 clips of the human study, 33 out of 50 had the identical set of four predictions ("turn-on tap", "turn-off tap", "open cupboard", "open drawer", which are the four most frequent actions in the dataset), and these slots used only 28 distinct actions across all 50 clips, compared to 196 that the baseline's top-5 used.

The cause of this issue is a mismatch between training targets and the model's ability. The coverage targets themselves are correctly conditional on the previous action, but the model recognises the previous action correctly only about 18% of the time. Thus, the model is trained toward targets it cannot reliably predict and falls back on the average of those targets. The result is that this model's diversity was fake.

Adding the grounding of Equation 6 with $\beta = 30$ fixes this. The cross-clip overlap value drops from 0.6-0.8 to 0.167, approaching the single head's level of 0.05, and object match nearly doubles from 27.8 to 52.2, meaning the extra hypotheses now name objects that are actually in the scene. The cost is that the set similarity returns to the baseline level of 0.207. This happens because concentrating predictions on the objects present in the scene makes the slots vary the verb instead of the noun.

Learning rate. The model with the object-compatibility grounding mechanism was first trained at the baseline's peak learning rate of 5×10^{-4}. The training was diverse for the first five epochs, then it collapsed when the warmup schedule reached its peak, resulting in exact duplicates reaching 56% at the end of training. Lowering the peak learning rate to 3×10^{-4} fixes this collapse. It results in a steady set similarity through the peak, and the model converges to its best values. The cost is a lower converged MT5R of 15.84, below the 16.53 of the model with no grounding, which is improved later by the noun-distinct selection. When retraining the baseline at 3×10^{-4}, it confirms that the lower learning rate does not give an advantage to the modified model.

Noun-distinct selection. With the grounding and learning rate fixed, switching on the noun penalty with $\rho = 100$ is the last step. Noun duplicates drop from 29.8% to 8.9% of pairs and set similarity from 0.207 to 0.142, below the baseline top-5, while diverse set recall is almost unchanged (from 22.49 to 22.13) and object match stays high at 43.4. The anchor slot also improves (MT5R increases from 15.84 to 16.59), and it seems like spreading the coverage targets over distinct objects reduces their interference with the anchor slot. The costs are that the verb duplicates are at the baseline level (19.4%), and object grounding is slightly weaker, which are both consequences of forcing one prediction per object.

In summary, the transition targets provide external information, grounding makes the predictions input-dependent, the lower learning rate makes training stable, the noun penalty makes the set diverse, and the deduplicated readout removes the rest of the duplicates.

5.5 Human Evaluation

The metrics measure diversity and plausibility with approximations, the similarity matrix, and the single annotated ground truth. A prediction can be plausible for a scene but count as wrong because a different action happened to follow. Therefore, this subsection adds a human evaluation and inspects the model's behaviour more directly.

Pilot human study setup. For 50 validation clips, both the baseline and the final modified model produced a prediction set of five actions. These clips were drawn uniformly at random from the validation split with a fixed seed, without stratification by participant, action class, or difficulty. Every individual predicted action was rated as plausible or not plausible given the observed clip (250 action ratings per model), and every five-action set was rated for diversity on a 1-5 scale (50 set ratings per model). The predictions equal to the ground-truth action were rated plausible, and an action predicted by both models for the same clip received the same rating in both sets. In addition, any action that was unclear whether it was plausible was rated as not plausible. The diversity ratings followed a specific rubric (start at 5, subtract half a point per pair sharing a verb or noun and one point per exact-duplicate pair, and adjust by at most one point for clip context). All ratings were given by the author, and the two prediction sets were shown side by side and labelled by model, so the ratings were not blind, making this a pilot evaluation. Table 8 reports the outcome.

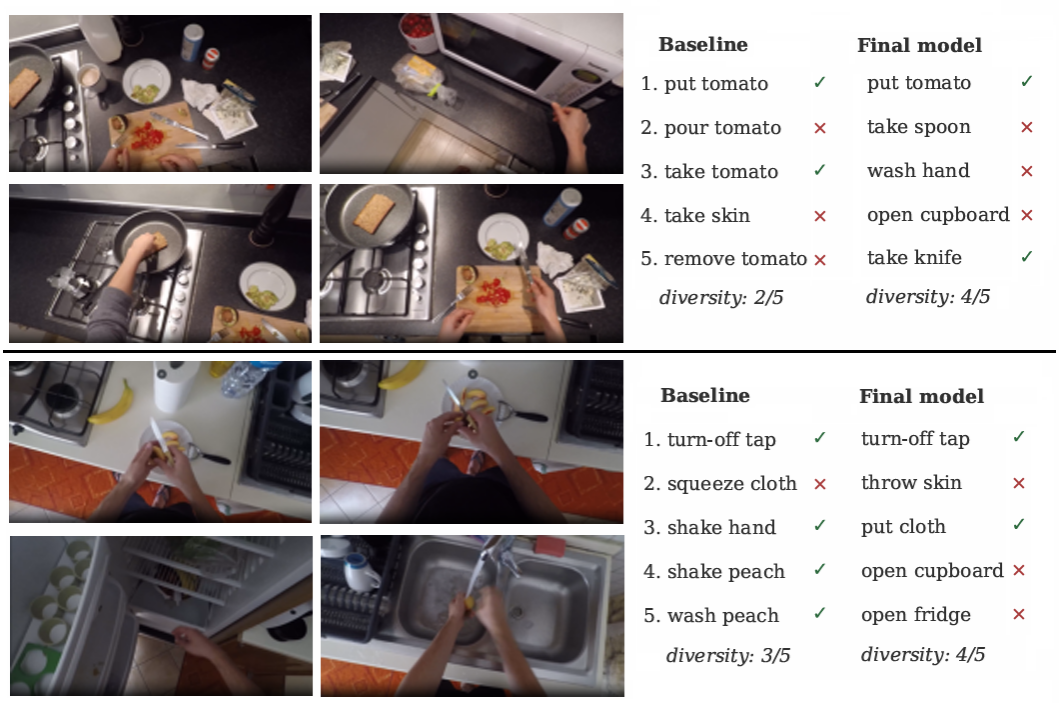

Figure 6: Qualitative comparison of two validation clips. For each clip, four frames were sampled at 0, 10, 20, and 32 seconds, with five predictions each from the baseline and the final model. Check marks and crosses indicate whether an action was annotated as plausible, and the rated 1-5 set diversity shows how diverse each set is. In the first clip, the baseline repeats "tomato" in four of five predictions (diversity 2/5), while the final model spreads its hypotheses across distinct objects (diversity 4/5). The second clip shows the same diversity gain, but also the model's errors. Some slots fall back on generic kitchen actions like "open cupboard" or "open fridge", which are present in the scene but are not that relevant to the activities happening.

Findings of the human study. On diversity, the final modified model's sets are rated 3.60 compared to the baseline's 2.90. Thus, the set-similarity metric difference between the two models agrees with the human ratings. The confidence interval of this per-clip difference does not include zero, so the diversity gain is unlikely to be caused by rating noise.

On plausibility, the human ratings show something that the automatic metrics do not measure. Per individual action, plausibility drops from 42.8% to 30.0%, or from 2.14 to 1.50 plausible actions per set. The confidence interval of this difference also does not include zero, so the performance drop is not explained by rating noise. The MT5R metric evaluates only the primary prediction, but the human study averages over all five, and forcing five different actions leads to diverse slots on less likely futures. This results in 24% more diversity and costs 30% relative per-action plausibility. Overall, the absolute plausibility is low for both models, as even the baseline has about 2 out of 5 actions annotated as plausible, which reflects the difficulty of one-second action anticipation on the EPIC-KITCHENS-100 dataset (MT5R around 16% and top-1 around 11%) and not a problem of either model, so the comparison should be read as relative.

Measure	Baseline	Final model	Difference (95% CI)
Plausible actions (of 250)	42.8%	30.0%	
Plausible actions per 5-set	2.14	1.50	-0.64 $[-1.02, -0.26]$
Mean set diversity (1-5)	2.90	3.60	$+0.70$ $[+0.42, +0.98]$

Table 8: Human evaluation on 50 validation clips, using the best checkpoint of each model. The final model uses deduplicated readout, and the baseline set uses its top-5. The final modified model's sets are more diverse, but have a per-action plausibility cost. The confidence intervals are computed on the per-clip differences between the two models.

6 Conclusions

The research question of this thesis was whether a next-action anticipation model can be modified to produce diverse yet plausible next-action predictions. To answer it, the baseline was extended from a single learnable query to a set of N parallel prediction slots, and a series of mechanisms were developed to make these predictions behave as intended.

Regarding the answers to the research question on diversity, the final model produces a wider variety of predictions than the baseline for the same set size, as set similarity and noun duplicates drop, and the human evaluation shows higher diversity for the final model against the baseline. However, reranking the baseline's output reaches the same diversity at a smaller recall cost and without retraining. The advantage of the trained model is that its predictions cover different objects while keeping the primary prediction unchanged. On plausibility, the anchor slot's class-mean top-5 recall matches the baseline's, so the benchmark score of the primary prediction is preserved. On the other hand, the whole set is less accurate, as the recall of the set is lower than the baseline's top-5, and in the human evaluation the fraction of predicted actions rated as plausible drops. The cost of diversity therefore falls only on the extra slots. Lastly, the trade-off between diversity and accuracy is controllable up to a specific set size, after which training becomes unstable.

6.1 Limitations

There are several limitations in this study. Firstly, evaluating diverse next action anticipation is difficult, because each validation clip has exactly one annotated next action, even though several different futures may be equally reasonable. As a result, diverse set recall punishes every prediction that is not the ground-truth next action, including predictions that an annotator would annotate as plausible for a scene. The human study compensates for this partially, but the metrics remain approximations, as they measure for one sampled future and not a set of possible futures.

Second, the similarity matrix S treats two actions as related only when they share a verb or a noun. This equation does not take into account semantic similarity between different action pairs. For example, "wash plate" and "rinse plate" count as only half-similar, and "wash plate" and "clean dish" as completely different pairs, even though they describe almost the same action.

Third, reranking the baseline's output reaches the same diversity at a lower recall cost than the trained mechanisms. The trained model performs better on object coverage, on grounding in the observed scene, and on the accuracy of its primary prediction, so the added training complexity is only justified when those properties matter.

A further limitation is that the coverage targets are conditioned on the last observed action, but the model recognises the previous action about 18% of the time. The object compatibility mechanism supports this issue, but the targets assume context information that the model can only partially extract from the clip. Beyond that, the transition prior is first-order, so it conditions on a single previous action and ignores the longer past context, and it is estimated purely from annotation statistics, without any visual input.

Next, the human evaluation covers 50 validation clips rated by one annotator, which limits its statistical strength. The ratings were not blind, as the annotator knew which model produced each prediction set, and with a single annotator, agreement between annotators could not be measured. A larger blind study with many independent annotators and agreement between them would make the plausibility and diversity findings more reliable. Furthermore, the confidence intervals only cover the variation across clips, but not the uncertainty of a single non-blind annotator. The reranked set was also not rated, so it is not known how the final model compares to reranking on plausibility.

Another limitation is that all experiments use EPIC-KITCHENS-100 with the frozen RULSTM RGB features. The mechanisms themselves are not kitchen-specific, but kitchen activity is highly sequential, which is what makes the transition prior informative, and the object compatibility mechanism relies on the noun vocabulary of the annotations. Whether the same modified model can be useful in less structured domains, other datasets, or other visual features has not been tested.

The diversity-accuracy trade-off analysis showed that training collapses at $N = 10$, with about half of the predictions being duplicates. The method, as presented, is therefore reliable only for moderate set sizes, and scaling to larger prediction sets would require modifications to ensure stability.

A further limitation is that every configuration was trained once, without repeated runs under different random seeds. Small differences between models, such as the gap between the final model's MT5R and the baseline's, could therefore also be due to run-to-run variation rather than to the modifications themselves.

Lastly, all development and reporting in this thesis used the same validation split. The checkpoints were selected on it, settings such as the learning rate were chosen based on it, and the final results are reported on it. Thus, the reported numbers are optimistic. The test split could not be used instead, since its labels are not public and its evaluation server returns only the official benchmark metrics, which do not include diversity measures. The comparisons remain informative, as every model was developed in the same way, so the ranking between models is more reliable than the exact values.

6.2 Further Research

This study could be continued and improved in different ways. Firstly, object compatibility is currently inferred from the nouns of past annotated actions, so an object that is visible but never interacted with by the human is invisible to the mechanism. An object detector running on the observed frames could check all the objects that are present, and therefore find which next actions are plausible in the scene.

Moreover, the transition prior could be extended from a single previous action, for instance by conditioning on the last few actions or by learning the transition model jointly with the anticipation model. In addition, improving the model's recognition of the previous action would improve the usefulness of the transition targets, since the targets would be conditioned on information the model actually has.

Another addition could be replacing the matrix S with a similarity derived from learned representations, as embeddings of the verb and noun labels would let both the diversity mechanisms and the metrics recognise that a word is close to another word. This change would make the noun-distinct selection and the set-similarity metric reflect meaning instead of the exact label.

Next, every configuration in this thesis was trained once. Training each of them with several random seeds and reporting the mean and spread across runs would show which of the measured differences are larger than the run-to-run variation. This matters most when the performance of two models is close, where a single run cannot separate a real effect from noise.

Finally, the human study could be scaled up with multiple raters and agreement statistics. The approach could also be tested on other egocentric datasets such as Ego4D [GWB+22], to conclude whether the final model's diversity mechanisms can transfer to other domains. This would require modifications to the model, since the baseline uses that dataset for long-term action anticipation.

References

[CMS+20] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *European Conference on Computer Vision (ECCV)*, pages 213–229, 2020.

[DDF+22] Dima Damen, Hazel Doughty, Giovanni Farinella, Antonino Furnari, Evangelos Kazakos, Jian Ma, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision: Collection, pipeline and challenges for EPIC-KITCHENS-100. *International Journal of Computer Vision*, 130:33–55, 2022.

[FF21] Antonino Furnari and Giovanni Maria Farinella. Rolling-unrolling LSTMs for action anticipation from first-person video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43:4021–4036, 2021.

[GAL+24] Hongji Guo, Nakul Agarwal, Shao-Yuan Lo, Kwonjoon Lee, and Qiang Ji. Uncertainty-aware action decoupling transformer for action anticipation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18644–18654, 2024.

[GD24] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *Conference on Language Modeling (COLM)*, 2024.

[GG21] Rohit Girdhar and Kristen Grauman. Anticipative video transformer. *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13505–13515, 2021.

[GWB+22] Kristen Grauman, Andrew Westbury, Eugene Byrne, et al. Ego4D: Around the world in 3,000 hours of egocentric video. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18995–19012, 2022.

[Kuh55] Harold W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

[MALL24] Himangi Mittal, Nakul Agarwal, Shao-Yuan Lo, and Kwonjoon Lee. Can't make an omelette without breaking some eggs: Plausible action anticipation using large video-language models. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18580–18590, 2024.

[QR25] Yihui Qiu and Deepu Rajan. Action sequence augmentation for action anticipation. *International Conference on Learning Representations (ICLR)*, 2025.

[RF24] D. Roy and B. Fernando. Predicting the next action by modeling the abstract goal. *International Conference on Pattern Recognition (ICPR)*, pages 162–177, 2024.

[RRF24] Debaditya Roy, Ramanathan Rajendiran, and Basura Fernando. Interaction region visual transformer for egocentric action anticipation. *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 6726–6736, 2024.

[TCP+26] Tsung-Ming Tai, Sofia Casarin, Andrea Pilzer, Werner Nutt, and Oswald Lanz. Action-guided attention for video action anticipation. *International Conference on Learning Representations (ICLR)*, 2026.

[TWL+20] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11662–11671, 2020.

[ZMS+26] Zeyun Zhong, Manuel Martin, David Schneider, David J. Lerch, Chengzhi Wu, Frederik Diederichs, Juergen Gall, and Jurgen Beyerer. Scalable video action anticipation with cross linear attentive memory. *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 8113–8123, 2026.

[ZWM+23] Zeyun Zhong, Chengzhi Wu, Manuel Martin, Michael Voit, Juergen Gall, and Jürgen Beyerer. DiffAnt: Diffusion models for action anticipation. *arXiv preprint arXiv:2311.15991*, 2023.