Supplementary Material of MUHACU

Yue Zhuo*†
terry.zhuo@unsw.edu.com
The University of New South Wales
Sydney, Australia

Yaqing Liao*
yaqingliao1997@gmail.com
University of Electronic Science and Technology of China
Chengdu, China

Yuecheng Lei*
leiyuecheng@gmail.com
University of Electronic Science and Technology of China
Chengdu, China

Lizhen Qu†
lizhen.qu@monash.edu
Monash University
Melbourne, Australia

Xiaojun Chang
cxj273@gmail.com
Monash University
Melbourne, Australia

Zenglin Xu
zenglin@gmail.com
Harbin Institute of Technology
Shenzhen, China

ACM Reference Format:
Yue Zhuo, Yaqing Liao, Yuecheng Lei, Lizhen Qu, Xiaojun Chang, and Zenglin Xu. 2021. Supplementary Material of MUHACU. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 DETAILS OF DATA COLLECTION

1.1 Crowdsourcing Annotations

- Action sequence evaluation: workers on AMT were given a video about human daily activities and seven sequences of actions for the video, which consisted of one ground truth and six "negative" answers. They were required to judge whether each action sequence is true or false according to the content of the video.
- intent inference: crowdsourcing experts were given a video about human daily activities and a sequence of actions for the video. They were required to answer a question like "The person has taken the above actions in the video because he/she wants to do what?".Their answers should be like the format of a good example, such as "eat a sandwich", "clean the kitchen" etc.
- intent and action sequence matching evaluation: workers on AMT were given a 20% initial video about human daily activities and six sequences of actions for the video, which consisted of one ground truth and five "negative" answers. They were required to judge whether each of the action sequences takes place in the last 80% of the video and matches the given intent.

1.2 Verification Scheme

In our procedure of dataset collection, we adopted a complete verification scheme referring to [2], which consisted of three main parts:

(1) Worker recruitment. We firstly published some test tasks and invited workers who were interested in our task to give us their email. Then we chose those workers according to their results of the test to send them an email containing our detailed instructions and explained the error of their results. In this way, we can confirm those workers really understand our task. Finally, our recruited workers would be allocated a qualification called "Master at Sequence Evaluation". Only workers who obtained the qualification could accept our task hits.
(2) Automatic pre-validation. We used two JavaScript validators to avoid workers randomly select answers or input results copied from our provided sequence.
(3) Human evaluation of collected data. For the task of intent inference, we introduced human experts as a third party to design a 5-point Likert scales to judge data from crowdsourcing experts, via three criteria: grammaticality, format, and semantic relevance. For the action sequence evaluation process and intent and action sequence matching evaluation process, we made a serial of rules for data collected from workers to get the final results. Then we randomly sampled 40 videos to ask human experts to give us their results. Finally, we adopted a quality consistency coefficient Kappa to evaluate the consistency between the results from workers and results from the third party.

In this section, we elaborate more on how to construct our dataset, and about our process of crowdsourcing

1.3 Action Sequences

We found that in charades a video may contain more than one activity, whereas we are more concerned with a single activity with a clear intent to do reasoning about future actions. Therefore, we design an activity (action sequence) extraction algorithm to organize the temporal activity chains from the action set of the charades video. This algorithm uses a greedy search that explores an action sequence by expanding the most promising action in a limited set. It uses text similarity, temporal relevance, and topic

*Authors contributed equally to this research.
†Corresponding author

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

matching between two actions to find the most possible next action in the candidate set. The details are shown below.

Definition 1. *(Normalized Action Phrase). To avoid the influence of irrelevant words, we extract verb and noun phrases from the original action class text descriptions and perform word lemmatization on them. (e.g. Washing their hands to wash hand, Someone is eating something to eat) . Normalized Action Phrase will be used for semantic similarity computation in the search step.*

Definition 2. *(Action Topic). We divide 157 actions into 22 different topics by object type (e.g. Opening a book and Closing a book are the same topic). The Action Topics will be used for the calculation of the topic matching score when searching for the next action.*

1. We first sort the action set of the video from early to late according to the start time.

2. The search is started from the earliest starting action after sorting. For the current action a_i, The candidate set contains the actions that have not been added to any action sequences and started after the start time of current action a_i. We calculate the scores $Score(a_i, a_j)$ of all candidates a_j according to the following equation,

$$\text{score}\left(a_i, a_j\right) = f_{\text{semantic}}\left(a_i, a_j\right) + f_{\text{time}}\left(a_i, a_j\right) + f_{\text{topic}}\left(a_i, a_j\right). \quad (1)$$

$$f_{\text{semantic}}\left(a_i, a_j\right) = \text{cosine}\left(E_{a_i}, E_{a_j}\right)$$
$$E_a = \sum_{w_i \in Txt^a} tfidf\left(w_i\right) * w2v\left(w_i\right), \quad (2)$$

$$f_{\text{time}}\left(a_i, a_j\right) = \left(1 - atanh(|S_t^{a_i} - S_t^{a_j}|\right) * \pi/2) \quad (3)$$

$$f_{\text{topic}}\left(a_i, a_j\right) = \mathbf{1}\left(Top^{a_i} = Top^{a_j}\right) \quad (4)$$

Where Txt_{a_i} is the normalized action phrase of action a_i, $S_t^{a_i}$ is the start time of a_i, and Top^{a_i} is the topic of a_i. Finally we select the action that has the highest score from the candidate set.

3. Compare the score of the selected most possible action with the relevance threshold. If the score is higher than the threshold, we consider this action as the next action and add it to the current action sequence. Then we can continue the searching, go back to step1 and sort the remaining actions that have not yet formed an action sequence. If there are no actions left, the search is done.

4. Otherwise, if the score is lower than the threshold, it means that all candidate actions are not very relevant to the current action, so the current action sequence search will be terminated. We then add the current sequence to the activity set and restart the search. Until all actions are added to an action sequence, the search is finished.

With this algorithm, Each video may contain multiple sequences of activities, consisting of related actions in temporal order, representing an event done by a person in the video. We first filter out 223 videos that don't have action set labels and 6622 videos that don't contain action sequences longer than 4, then sample one action sequence for each video. After filtering 3003 videos are left, each with a sampled action sequence. These examples contain 2371 non-repetitive action sequences, covering 148 of the 157 action classes.

1.4 Experts Annotation

As mentioned in our paper, our dataset firstly needs some experts to annotate the intent of a video based on the content of the video and an action sequence related to the video.

We use Amazon Mechanical Turk (AMT) for expert annotation. A screenshot of our interface of this task is given in Figure 6. Given a full video and a related sequence of actions, experts answer a concrete intent and a high-level intent. The high-level intent should be generalized from the concrete intent. These are all written in the format "verb + something".For example, if the concrete intent is "drink some water", then a high-level intent "satisfy his thirst" may be right. In our annotation User Interface (UI), experts should write their answers in the input area which gives an instruction "write down the intent".

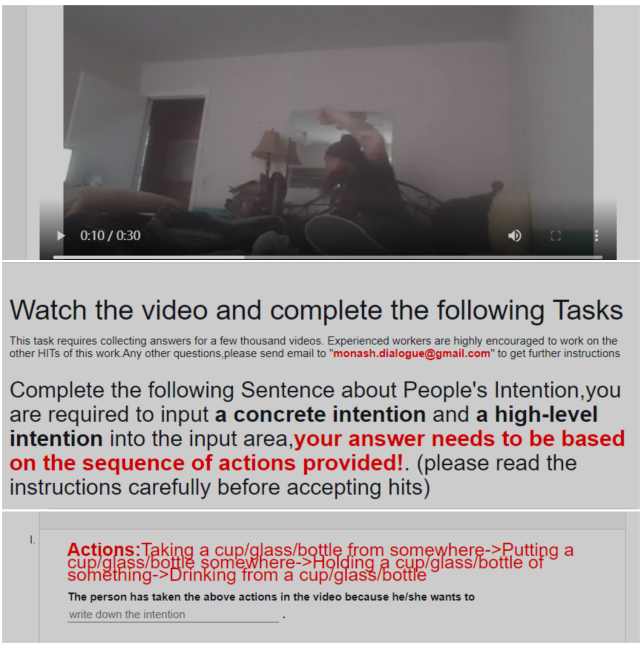

Figure 1: A screenshot of experts annotation interface

1.5 Crowdsourcing and Quality Data

Planning crowdsourcing. When we get a high-level intent of each video, we could continue designing our planning crowdsourcing task, which gives workers 20% of the initial video and the related action sequence and the high-level intent of the full video. Workers should do some judgment on six action sequences based on the above information. They only need to choose "True" or "False" of each action sequence. A screenshot of the planning task is given in figure 7.

Forecasting Crowdsourcing. The mainframe is similar to planning to crowdsource, but the high-level intent of a video is removed. That is to say, we hope workers select the most proper future action sequence according to 20% of the initial video and the related action sequence. A screenshot of the forecasting task is given in figure 8.

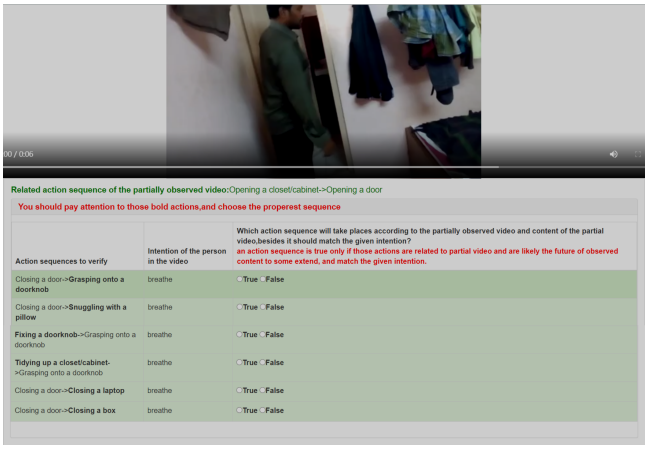

Figure 2: A screenshot of planning task crowdsourcing interface

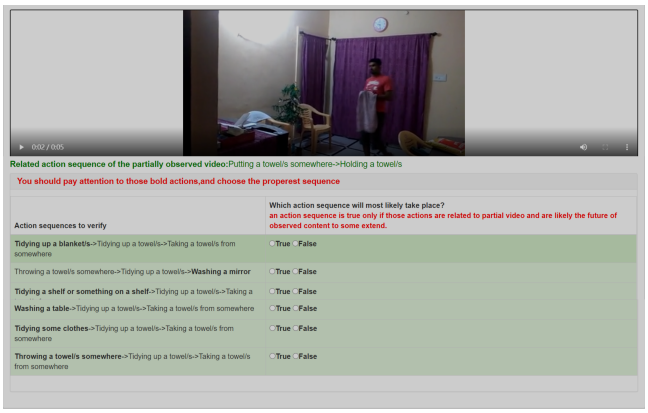

Figure 3: A screenshot of forecasting task crowdsourcing interface

Quality Data. Automated quality checks: We added some JavaScript validator checks to the crowdsourcing UI to ensure high quality. To prevent workers from selecting choices before finishing watching our videos, we set a default limit JavaScript validator, which enables the selection area after the video plays to the end. Another JavaScript validator that restricts workers from select more than one true sequence helps us using some rules to obtain our crowdsourcing results. Our rules are showing below.

Rules 1. *1.we set five hits for each video, that is to say, we will get five results from five workers.*

Rules 2. *2.If more than four of the five results give the same answer, then we adopt the same answer as this sequence's final result. for example, if there is a sequence that at least four workers select the "True(or False)" option, then we think this sequence is "True(or False)".*

Rules 3. *3.After the above filtering rule, only three of the five results give the same answer. Then we introduce a third-party human check to judge whether the sequence is true or false. The third party (human expert) should review our videos, then combine the remaining sequences with the results from workers, finally give their judgment.*

Setting		Third party		sum
		True	False	
Rules	True	34	3	37
	False	4	199	203
sum		38	202	240

Table 1: Detail of Kappa at action sequence evaluation task

Setting		Third party		sum
		True	False	
Rules	True	37	3	40
	False	3	197	200
sum		40	200	240

Table 2: Detail of Kappa at intent and action sequence matching evaluation task

Rules 4. *4.After the third party finishing their works, we will get most videos that only have one true sequence and five false sequences. We adopt these results of videos as our final dataset. As for others that maybe have more than one true sequence or zero true sequence, we check these videos and decide whether the results of these videos are proper to our task. For the planning task, we only choose the properest sequence as a single choice, so we will remove those videos with zero true sequences and only select one true sequence as a final true sequence for those videos with more than one true sequence. For the forecasting task, we also remove those videos with zero true sequences. Whereas there exist too many videos with more than one true sequence, so we decide to divide the forecasting test dataset into the single choice part and multiple choices part.*

Results Consistency. In order to judge our crowdsourcing results can be used to construct our dataset, we adopt the Kappa coefficient as a measure for planning and forecasting sequence evaluation task. We randomly select 40 videos to ask the third-party experts to finish them just like crowdsourcing workers and get scores of 0.9100 and 0.8885 respectively. There detailed Kappa of the two evaluation tasks is shown in Table 12 and Table 13 respectively.

2 DETAILS OF RETRIEVAL AND SCORING METHOD

2.1 State Generation Algorithm

Algorithm 1: StateGenerator

Input: Action sequence $As = a_1, a_2, \ldots, a_n$, and symbolic components $\langle pre^{a_i}, add^{a_i}, del^{a_i} \rangle$ for each action a_i, where pre^{a_i}, add^{a_i}, del^{a_i} are sets of states

Output: init states S_{init}, final states S_{final}

$S_{init} = pre^{a_1}$;
$S_{Add} = add^{a_1}$;
for $i = 2; i \leq n; i++$ **do**
  $CurPre = pre^{a_i}$;
  for *state* $s \in CurPre$ **do**
    if $s \notin S_{init}$ *and* $s \notin S_{Add}$ **then**
      Append s to S_{init};
    end
  end
  Extend add^{a_i} to S_{Add};
end
$S_{cur} = S_{init}$;
for $j = 1; j \leq n; j++$ **do**
  $CurAdd = add^{a_j}$;
  $CurDel = del^{a_j}$;
  for *state* $s \in CurDel$ **do**
    if $s \in S_{cur}$ **then**
      Remove s from S_{cur};
    end
  end
  Extend $CurAdd$ to S_{cur};
end
$S_{final} = S_{cur}$;
return S_{init}, S_{final};

3 DETAILS OF END-TO-END METHOD

3.1 Training Details

The baselines are trained on 2 Nvidia Tesla V100 GPUs during UniVL and ProphetNet fine-tuning. For each task, UniVL and ProphetNet are run 50 epochs respectively based on the slight-modified default configurations reported on the original experiment. We constantly use 20% part of the video clip of our input visual features and the remained 80% part as the desired output ground truth. The initial training dataset containing 2402 Charades videos is distributed by the standard split 8 : 2 for training (1921 videos) and validation (481 videos).

4 END-TO-END TRAINING DISCUSSION

To ease the explanation, we define the setting of using the original discrete word representation action labels as the word-level input while the action-level input can be elaborated as those concatenated action labels in the modified vocabulary dictionary of the encoder and decoder. We conduct the experiment in these two kinds of text input during the UniVL training. Consequently, ProphetNet will learn two different level inputs accordingly. In order to control the generation quality from ProphetNet, we limit the output to constantly be the action-level by reducing the probability of out-of-distribution (OOD) action classes. OOD action classes are typically out of Charades 157 action classes and hence makes the evaluation of future action prediction unreliable.

As shown in Table 3, action-level input setting performs evenly better than word-level input setting on our two tasks. Experimentally, we argue that the action-level setting is more suitable in this kind of scenario. In the word-level setting, Sequence-to-Sequence models are more likely to apply the knowledge from the pre-trained models and therefore generate uncontrollable tokens when completing a single action representation. By regulating the input setting under action-level, the models should learn the dependence between action tokens better without any other previous biases.

Table 3: Comparison of Top 10 generation results between Acton-level setting and Word-level setting

task	setting	Quality						Diversity	
		precision	recall	seq-item-acc	seq-hit-rate	BLUE1	BLUE2	Dist1	Dist2
Action Forecasting	Action-level	36.77	28.66	10.68	3.42	27.34	12.60	17.77	37.01
	Word-level	17.48	14.67	3.73	0.00	14.51	5.42	19.75	39.53
Action Planning	Action-level	38.71	30.73	11.06	5.79	29.60	13.71	15.45	35.37
	Word-level	18.99	16.10	4.03	0.00	16.04	5.74	15.93	35.83

Table 4: Fact Threshold Controlling

Fact Threshold	Retrieval-Scoring Top10 (Top5)			Generation-Scoring Top10 (Top5)		
	Soft Acc.(%)	Precision.(%)	Recall.(%)	Soft Acc.(%)	Precision.(%)	Recall.(%)
0.9	62.67 (61.33)	59.44 (59.11)	32.67 (31.33)	74.67 (73.33)	72.67 (73.33)	39.33 (36.67)
0.8	65.33 (62.67)	59.44 (58.44)	34.00 (32.00)	76.00 (73.33)	72.67 (72.67)	40.00 (37.33)
0.7	66.67 (62.67)	59.22 (58.44)	34.67 (32.00)	76.00 (74.67)	69.33(72.67)	41.33 (38.67)
0.6	69.33 (62.67)	59.22 (58.44)	36.00 (32.00)	78.67 (74.67)	67.33 (72.67)	44.67 (40.00)
0.5	72.00 (65.33)	59.00 (59.56)	37.33 (33.33)	81.33 (74.67)	67.11 (72.00)	47.33 (40.00)
0.4	74.67 (65.33)	59.00 (58.22)	38.67 (33.33)	81.33 (76.00)	66.89 (72.00)	48.00 (40.67)
0.3	74.67 (66.67)	58.78 (58.89)	39.33 (34.67)	82.67 (77.33)	67.44 (70.67)	48.67 (41.33)
0.2	74.67 (66.67)	55.00 (56.89)	40.67 (34.67)	85.33 (77.33)	68.56 (70.00)	50.00 (42.67)
0.1	76.00 (68.00)	53.67 (52.88)	44.00 (36.00)	85.33 (77.33)	67.78 (68.67)	50.67 (42.67)

Table 5: Comparison of Retrieval-based method and Generation-based method and Human Baseline

Method	Action Forecasting-single		Action Forecasting-multiple				Action Planning	
	Precision(%)	Var	Soft Precision(%)	Precision (%)	Recall (%)	Var	Precision(%)	Var.(%)
Retrieval-Scoring Top10(Top5)	56.55 (55.86)	-	65.33	59.55	33.33	-	62.35 (60.98)	-
Generation-Scoring Top10(Top5)	65.28 (64.36)	-	74.66	72.00	40.00	-	60.19 (59.80)	-
Human Baseline	**65.33**	0.0697	**94.25**		-	0.1106	**67.00**	0.1548

5 DETAILS OF DISCUSSION

5.1 Comparison Details of RQ1

- **The major difference between retrieval-based method and end-to-end learning method.** The end-to-end learning method is a generation-based method, which needs supervised training of the model on the data of KB and test on the evaluation tasks. However, the retrieval-based method is a two-stage unsupervised method, using search strategies and similarity measures to find the most related support set in KB and score the answers.The advantages of the retrieval-based method are better explainability and diversity. However, it has the disadvantage that the quality of the top-k retrieval results are easily affected by action recognition noise. And it can only retrieve existing action sequences

Table 6: Comparison of action recognition methods

Method	Action Recognition			
	Acc@1	Acc@5	Rec@1	Rec@5
prototype	21.96	18.84	9.09	34.73
slowfast	34.48	21.77	13.34	39.69
i3d	36.22	24.35	14.83	45.13

from the knowledge base, whereas the end-to-end generation model can generate action sequences that are not in the knowledge base.

- **Comparison of knowledge base retrieval with top-K prediction of End-to-End models** We compare the top-k sequences obtained by the retrieval-based and generation-based methods. The distinct metric shows that the retrieval results are more diverse than the generation results in both tasks. For the action planning task, the quality of the top-k retrieval results is better than the top-k generation results in the quality metrics, but in the forecasting task, the top-k generation results are better than the top-k retrieval results. This is mainly due to the fact that the results of retrieval are easily affected by the quality of the query, i.e. the noise of the action recognition. In contrast, retrieval results with strong noise can be filtered by the intent matching in the planning task.
- **Comparison of QA evaluation results between retrieval model and generation model** It has been shown in the Table 5 that when passing the top-k retrieved activity facts and top-k generation action sequences to the same scoring function, the results of the QA evaluation are consistent with the quality of the top-k facts. The retrieval-based model outperforms the generation-based model in the task of action planning, but do not perform well in the forecasting task.
- **Top-k Fact Threshold Controlling** We report the process of selecting the best performances of the retrieval-scoring and generation-scoring methods in Human Action Forecasting (multiple) task based on the top-10 and top-5 fact score threshold as shown in Table 4. We take 0.1 for each step since the fact score is normalized in the range of $[0, 1]$.

5.2 Comparison Details of RQ2

- **Impact of action recognition.**
  There are two ways to do the retrieval using our multi-model knowledge base, one directly uses the low-level visual features of the input video as a query to do similar video retrieval, while the other first recognize the high-level video contents (actions) then utilize the text retrieval method to retrieve the most relevant action sequences and their corresponding videos. The comparison of these schemes is shown in Table 7.
  As for the first method, we compare video-level feature search method using cosine distance between the video-level feature of the query video and the video-level features of candidates to score the facts in knowledge base, and snippet-level feature alignment method OTAM to compute the similarity between snippet-level feature sequences. In another method, we compare the Prototype_BM25, i3d_BM25, and SlowFast_BM25 to do the action recognition. In the Prototype_BM25, we use the mean-pooling vector of input video as the query, to search in the 157 visual prototypes from KB to find the top-5 neighbors. And in i3d_BM25 and SlowFast_BM25 approaches, an action probability distribution can be obtained by i3d and SlowFast[1], then we filter the actions in top-5 by the threshold. The textual action set will be set as the query of the BM25.
  In this work, we find better leveraging recognition models can improve retrieval results in all metrics when compared to method only use the low-level feature search. We also report the results obtained by using the ground truth observed actions as the query of BM25, which can be served as a soft upper bound for the retrieval method using action recognition and textual retrieval. We find that when using the ground truth observed actions, the result of the retrieval module has a significant improvement. Therefore, the accuracy of the recognition model has a great influence on the final result. Table 6 shows a comparison of different action recognition methods on the input video clips of the evaluation task.
- **Usefulness of re-ranking by visual features and forecasting rules.** In the re-ranking stage of retrieval-based method, we use visual alignment scores combined with original BM25 scores to re-rank the top-50 results of BM25-initial retrieval. Visual alignment scores are computed between the snippet-level visual feature sequences of the query video and the candidate action sequences. It is shown in Table 2 that using visual features to re-ranking the results performs better than the methods that only use BM25 to rank the retrieved results.
  We also observe that adding some forecasting rules can help the process of re-ranking. The results in 7 show that the forecasting rules can also improve the recall, seq-item-acc and BLEU score. Especially under the assumption that the action recognition is completely correct, adding this rule score can significantly improve the hit rate.
- **Effectiveness of high-level intent.** In this set of experiments, we study the effect of adding intent in the retrieval process. It has been shown in Table 7 that leveraging high-level intent can improve the quality of retrieved results. This might be due to lack of discriminative action information for the partially observed videos and the error rate of initial action recognition. When adding the high level intent as the goal of input video, the possible actions in the future will become more certain, making the search results more accurate.

REFERENCES

[1] Haoqi Fan, Yanghao Li, Bo Xiong, Wan-Yen Lo, and Christoph Feichtenhofer. 2020. PySlowFast. https://github.com/facebookresearch/slowfast.
[2] Jekaterina Novikova, Oliver Lemon, and Verena Rieser. 2016. Crowd-sourcing nlg data: Pictures elicit better data. *arXiv preprint arXiv:1608.00339* (2016).

Table 7: Study of knowledge base retrieval

	Method	Quality						Diversity	
		precision	recall	seq-item-acc	seq-hit-rate	BLUE1	BLUE2	Dist1	Dist2
Methods using low-level visual features only	video-level feature search	22.76	22.40	5.58	8.55	22.43	9.31	58.70	88.11
	snippet-level feature alignment	21.99	22.02	4.92	11.11	22.08	9.11	57.88	87.97
Methods using action recognition and text retrieval	Prototype_BM25	29.14	25.03	7.47	18.80	22.69	9.73	19.21	55.09
	SlowFast_BM25	29.51	24.52	8.00	19.66	22.21	9.74	23.76	59.08
	i3d_BM25	30.29	25.31	8.11	22.22	22.85	10.02	23.22	57.92
	*gt_BM25	77.80	66.87	22.49	69.23	59.64	28.99	20.03	55.59
Methods adding re-ranking	Prototype_BM25 + OTAM	29.59	25.28	8.01	26.50	22.90	10.19	19.36	54.48
	Prototype_BM25 + OTAM_Rule	29.67	25.41	8.19	27.35	23.08	10.37	20.40	55.31
	i3d_BM25 + OTAM	30.89	25.85	8.95	28.21	23.26	10.76	23.75	57.22
	i3d_BM25 + OTAM_Rule	30.88	25.89	9.18	29.91	23.28	10.93	23.70	57.10
	*gt_BM25 + OTAM_Rule	77.13	66.93	24.37	82.91	59.97	30.45	20.35	54.54
Methods adding intent aligning	Prototype_BM25_Intention + OTAM_Rule	39.65	**39.80**	10.32	33.88	**38.73**	**17.36**	32.86	69.96
	i3d_BM25_Intention + OTAM_Rule	**39.91**	35.48	**11.11**	**36.36**	33.87	15.70	33.05	67.41
	*gt_BM25_Intention + OTAM_Rule	73.90	64.86	24.00	77.69	59.34	29.83	20.90	56.07

[1] Method beginning with *gt means that it uses ground truth observed actions as the input of BM25 search, which can be considered as upper bounds for methods using action recognition.