Zero-Shot Anticipation for Instructional Activities

Fadime Sener
University of Bonn, Germany
sener@cs.uni-bonn.de

Angela Yao
National University of Singapore
ayao@comp.nus.edu.sg

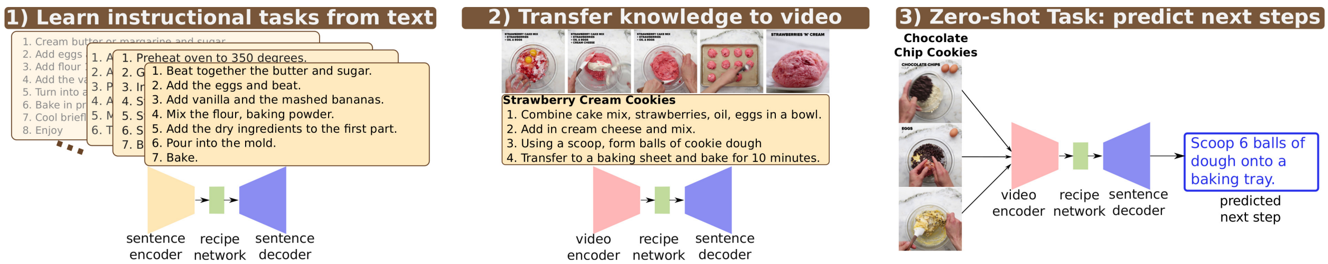

Figure 1: We learn procedural knowledge from large text corpora and transfer it to the visual domain to anticipate the future. Our system is composed of four RNNs: a sentence encoder and decoder, a video encoder and a recipe network.

Abstract

How can we teach a robot to predict what will happen next for an activity it has never seen before? We address this problem of zero-shot anticipation by presenting a hierarchical model that generalizes instructional knowledge from large-scale text-corpora and transfers the knowledge to the visual domain. Given a portion of an instructional video, our model predicts coherent and plausible actions multiple steps into the future, all in rich natural language. To demonstrate the anticipation capabilities of our model, we introduce the Tasty Videos dataset, *a collection of 2511 recipes for zero-shot learning, recognition and anticipation.*

1. Introduction

Imagine a not-so-distant future, where your kitchen is serviced by a robot chef[1]. How should we teach robots to cook? By reading all the recipes on the web? By watching all the cooking videos on YouTube? The ability to learn and generalize from a set of instructions, be it in text, image, or video form, is a highly challenging and open problem faced by those working in machine learning and robotics.

In this work, we limit our scope of training the next robo-chef to predicting subsequent steps as it watches a human cook a never-before-seen dish. We frame our problem as one of future action prediction in a zero- and/or few-shot learning scenario. This best reflects the situation under which service robots will be introduced [18, 49]. The robot is pre-trained extensively, but not necessarily with knowledge matching exactly the deployment environment, thereby forcing it to generalize from prior knowledge. At the same time, it is important for the robot to anticipate what will happen in the future, to ensure a safe and smooth collaborative experience with the human [28, 56].

Instructional data and in particular cooking recipes can be readily found on the web [1, 2]. The richest forms are multimodal, *e.g.* images plus text, or videos with narrations. Such data fits well into our scenario in which the service robot visually recognizes the current context and makes future predictions. However, learning complex, multi-step activities requires significant amounts of data, and despite their online abundance, it is still difficult to find sufficient examples in multi-modal form. Furthermore, learning the visual appearance of specific steps would require temporally aligned data, which is less common and/or expensive to obtain. Our strategy is therefore to separate the procedural learning from the visual appearance learning. Procedural knowledge is learned from text, which is readily available in large corpora on the scale of millions [46]. This knowledge is then transferred to video, so that the learning of visual appearances can then be simplified to only a grounding model done via aligned video and text (Fig. 1). More specifically, we encode text and/or video into context vectors. The context is fed to a recipe network, which models the sequential structure of the recipe and makes following step predictions in vector form which are then decoded back into sentences.

Our work is highly novel in two key regards. First and foremost, we are working with zero-shot action anticipation under a semi-supervised setting, as we target prediction for never-before-seen dishes. We achieve this by generalizing cooking knowledge from large-scale text corpora and then transferring the knowledge to the visual domain. This relieves us of the burden and impracticality of provid-

[1] Robots cooking specific recipes [3, 9, 51] already exist!

ing annotations for a domain in which there are virtually unlimited number of categories (dishes) and sub-categories (instructional steps). We are the first to tackle such a problem in this form; prior works in complex activity recognition are severely limited in the number of categories and steps [6, 29, 30, 43], while works in action anticipation rely on strong supervision [5, 31, 61].

Second, we do not work with closed categories derived from word tags; instead we train with and also predict full sentences, *e.g*. *'Cook the chicken wing until both sides are golden brown.'* vs. *'cook chicken'*. This design choice makes our problem significantly more challenging, but also offers several advantages. First of all, it adds richness to the instruction, since natural language conveys much more information than simple text labels [32, 59]. It also allows for anticipation of not only actions but also objects and attributes. Finally, as a byproduct, it facilitates data collection, as the number of class-based annotations grows exponentially with the number of actions, objects and attributes and leads to very long-tailed distributions [16].

When transferring knowledge from text recipes to videos, we need to ground the two domains with video with temporally aligned captions. To the best of our knowledge, YoucookII [59] is currently the only dataset with such labels. However, it lacks diversity in the number of dishes and therefore unique recipe steps. As such, we collect and present our new *Tasty Videos dataset*, a diverse set of 2511 different cooking recipes[2] accompanied by a video, ingredient list, and temporally aligned recipe steps. Video footage is taken from a fixed birds-eye view and focuses almost exclusively on the cooking instructions, making it well-suited for understanding the procedural steps.

We summarize our main contributions as follows:

- We are the first to explore zero-shot action anticipation by generalizing knowledge from large-scale text-corpora and transferring it to the visual domain.
- We propose a modular hierarchical model for learning multi-step procedures with text and visual context.
- Our model generalizes cooking knowledge and is able to predict coherent and plausible instructions for multiple steps into the future. The predictions, in rich natural language, score higher in standard NLP metrics than video captioning methods which actually observe the visual data on YouCookII and Tasty Videos.
- We demonstrate how the proposed approach can be useful for making future step predictions in a zero-shot scenario compared to a supervised setting.
- We present a new and highly diverse dataset of 2511 cooking recipes which will be made publicly available and be of interest for those working in anticipation, complex activity recognition and video captioning.

[2] Collected from the website `https://tasty.co/`

2. Related Works

Understanding complex activities and their sub-activities has been addressed typically as a supervised video segmentation and recognition problem [29, 40, 43]. Newer works are weakly-supervised, using cues from narrations [34, 48, 6] or receiving ordered sequences of the actions in videos [11, 24, 41], or fully unsupervised [47]. Our work is similar to those using text cues; however, we do not rely on aligned visual-text data for learning the activity models [6, 48] but rather for grounding visual data.

Action prediction is a new and fast-growing area. Methods for early event recognition [45, 23, 57] are sometimes (confusingly) also referred to as action prediction, but are incomplete inference methods, since a portion of the action has been observed. Prior work in forecasting activities before making *any* observations have been limited to simple movement primitives [28], or personal interactions [31, 55]. Single predictions are made and the anticipated actions typically occur within a few seconds time frame. Recently, [5] predicts multiple actions into the future; our method also predicts multiple steps but unlike [5], we do not require repetitions of activity sequences for training.

The cooking domain is popular in NLP research, since recipes are rich in natural language yet are reasonably limited in scope. Modelling the procedural aspects of text and generating coherent recipes span several decades of work [15, 19, 25, 36, 37]. In multimedia, recipes are involved in tasks such as food recognition [21], recommender systems [35] and indexing and retrieval [12, 46]. In computer vision, cooking has been well-explored for complex and fine-grained activity recognition [30, 43, 17, 42, 16, 59], temporal segmentation [30, 59] and captioning [44, 39, 60]. Several cooking and kitchen-related datasets have been presented [16, 34, 43, 30, 59] and feature a wide variety of labels depending on the task. Two [34, 59] are similar to our new dataset, in that they include recipe texts and accompanying videos. However, YouCookII [59] has limited diversity in activities with only 89 dishes; [34] is larger in scale, but lacks temporal alignments between texts and videos.

3. Modelling Sequential Instructions

Sequence-to-sequence learning [50] has made it possible to successfully generate continuous text and build dialogue systems [13, 54]. Recurrent neural networks (RNNs) are used to learn rich representations of sentences [22, 7, 27] in an unsupervised manner, using the extensive amount of text that exists in books and web corpuses. However, for instructional text such as cooking recipes, such representations tend to do poorly, and suffer from coherence from one time step to the next, since they do not fully capture the underlying sequential nature of the instruction set. As such, we propose a hierarchical model with four components, where the sentences and the steps of the recipe are

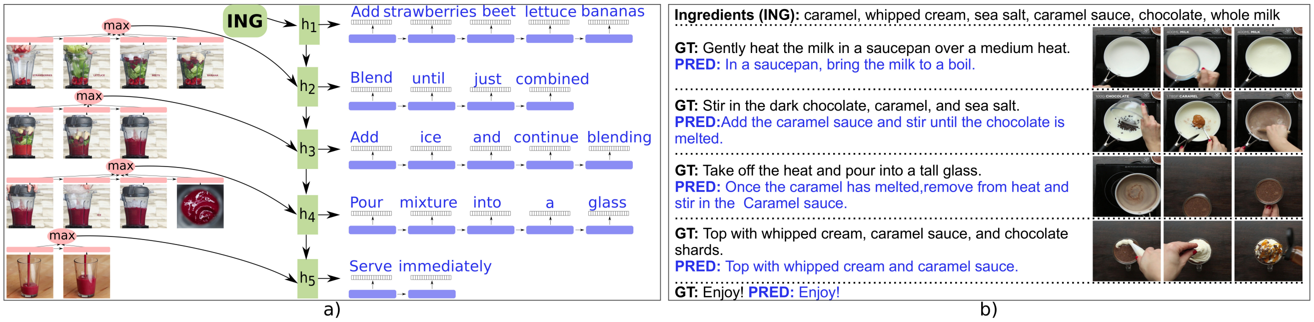

Figure 2: Left: our visual model, composed of video encoder, sentence decoder and recipe RNN. Given the ingredients as initial input and context in visual form, the recipe RNN predicts future steps decoded back into natural language. Right: next step prediction of our visual model. The blue sentences are our model's predictions. Note that our model predicts the next steps before seeing these segments!

represented by two dedicated RNNs: the sentence encoder and the recipe RNN respectively. A third RNN decodes predicted recipe steps back into sentence form for human-interpretable results (sentence decoder). These three RNNs are learned jointly as an auto-encoder in an initial training step. A fourth RNN encoding visual evidence (video encoder) is then learned in a subsequent step to replace the sentence encoder to enable interpretation and future prediction from video. An overview is shown in Fig. 1, while details of the RNNs are given in Sections 3.1 to 3.3.

3.1. Sentence Encoder and Decoder

The sentence encoder produces a fixed-length vector representation of each recipe step. We use a bi-directional LSTM and following [14] we apply a max pooling over each dimension of the hidden units. More formally, let sentence s_j from step j of a recipe (we assume each step is one sentence) be represented by M words, *i.e.* $s_j = \{w_j^t\}_{t=1...M}$ and $\mathbf{x}_j^t$ be the word embedding of word w_j^t. For each sentence j, at each (word) step t, the bi-directional LSTM_{se} outputs $\mathbf{y}_j^t$, where

$$\mathbf{y}_j^t = \left[\text{LSTM}_{\text{se}}\left(\{\mathbf{x}_j^1, ..., \mathbf{x}_j^t\}\right), \text{LSTM}_{\text{se}}\left(\{\mathbf{x}_j^M, ..., \mathbf{x}_j^t\}\right)\right] \quad (1)$$

which is a concatenation of the hidden states from the forwards and backwards pass of LSTM_{se}. The overall sentence representation $\mathbf{r}_j$ is determined by a dimension-independent max-pooling over the time steps, *i.e.*

$$(\mathbf{r}_j)_d = \max_{t \in \{1,...,M\}} (\mathbf{y}_j^t)_d, \quad (2)$$

where $(\cdot)_d,\ d \in \{1, ..., D\}$ indicates the d-th element of the D-dimensional bi-directional LSTM outputs $\mathbf{y}_j^t$. The decoder is an LSTM-based neural language model which converts the fixed-length representation of the steps back into sentences. More specifically, given the prediction $\hat{\mathbf{r}}_j$ from the recipe RNN of step j, it decodes the sentence $\hat{s}_j$

$$\hat{s}_j = \text{LSTM}_{\text{d}}(\hat{\mathbf{r}}_j) = \{\hat{w}_j^1, ..., \hat{w}_j^{\hat{M}}\}. \quad (3)$$

3.2. Recipe RNN

We model the sequential ordering of recipe steps with an LSTM which takes as input $\{\mathbf{r}_j\}_{j=1,...,N}$, *i.e.* fixed-length representations of the steps of a recipe with N steps, where j indicates the step index. At each (recipe) step, the hidden state of the recipe RNN $\mathbf{h}_j$ can be considered a fixed-length representation of all recipe steps $\{s_1, ..., s_j\}$ seen up to step j; we directly use this hidden state vector as a prediction of the sentence representation for step $j+1$, *i.e.*

$$\hat{\mathbf{r}}_{j+1} = \mathbf{h}_j = \text{LSTM}_{\text{r}}(\{\mathbf{r}_0, ..., \mathbf{r}_j\}). \quad (4)$$

The hidden state of the last step $\mathbf{h}_N$ can be considered a representation of the entire recipe. Due to the standard recursion of the hidden states in LSTM_{r}, each hidden state vector and therefore each future step prediction is conditioned on the previous steps. This allows to predict recipe steps which are plausible and coherent with respect to previous steps.

Recipes usually include an ingredient list which is a rich source of information that can also serve as a strong modelling cue [25, 46]. To incorporate the ingredients, we form an ingredient vector $\mathbf{I}$ for each recipe in the form of a one-hot encoding over a vocabulary of ingredients. $\mathbf{I}$ is then transformed with a separate fully connected layer in the recipe RNN to serve as the initial input, *i.e.* $\mathbf{r}_0 = f(\mathbf{I})$.

3.3. Video Encoder

For inference, we would like the recipe RNN to interpret sentences from text inputs and also visual evidence. Due to the modular nature of our proposed model, we can conveniently replace the sentence encoder with an analogous video encoder. Suppose the j^{th} video segment c_j is composed of L frames, *i.e.* $c_j = \{\mathbf{f}_j^t\}_{t=1,...,L}$. Each frame f_j^t is represented as a high-level CNN feature vector – we use the last fully connected layer output of ResNet-50 [20] before the softmax layer. Similar to the sentence encoding $\mathbf{r}_j$ in Eqs. 1 and 2, we determine the video encoding vector $\mathbf{v}_j$ by applying a dimension-independent max pooling over the time steps of $\mathbf{z}_j^t$, where :

$$\mathbf{z}_j^t = \left[\text{LSTM}_{\text{ve}}\left(\{\mathbf{f}_j^1, ..., \mathbf{f}_j^t\}\right), \text{LSTM}_{\text{ve}}\left(\{\mathbf{f}_j^M, ..., \mathbf{f}_j^t\}\right)\right]. \quad (5)$$

The video encoding LSTM_{ve} is trained such that $\mathbf{v}_j$ can directly replace $\mathbf{r}_j$, as detailed in the following.

3.4. Model Learning and Inference

The full model is learned in two stages. First, the sentence encoder (LSTM_{se}), recipe RNN (LSTM_{r}) and sentence decoder (LSTM_{d}) are jointly trained end-to-end. Given a recipe of N steps, a loss can be defined as the negative log probability of each reconstructed word:

$$L(s_1, ..., s_N) = -\sum_{j=1}^{N}\sum_{t=1}^{M_j} \log P(w_j^t | w_j^{t'<t}, \hat{\mathbf{r}}_j), \quad (6)$$

where $P(w_j^t | w_j^{t'<t}, \hat{\mathbf{r}}_j)$ is parameterised by a softmax function at the output layer of the sentence decoder to estimate the distribution over the words w in our vocabulary V. The overall objective is then summed over all recipes in the corpus. The loss is computed only when the LSTM is learning to decode a sentence. This first training stage is unsupervised, as the sentence encoder and decoder and the recipe RNN require only text inputs which can easily be scraped from the web without human annotations. In a second step, we train the video encoder (LSTM_{ve}) while keeping the recipe RNN and sentence decoder fixed. We simply replace the sentence encoder with the video encoder while applying the same loss function as defined in Eq. 6. This step is supervised, as it requires video segments of each step temporally aligned with the corresponding sentences.

During inference, we provide the ingredient vector $\mathbf{r}_0$ as an initial input to the recipe RNN, which then outputs the predicted vector $\hat{\mathbf{r}}_1$ for the first step (see Fig. 2). We use the sentence decoder and generate the first sentence $\hat{s}_1$. Then, we sample a sequence of frames from the video and apply the video encoder to generate $\mathbf{v}_1$ which we again provide as input to the recipe RNN. The output prediction of the recipe RNN, $\hat{\mathbf{r}}_2$, is for the second step of the video. We again use the sentence decoder and generate the next sentence $\hat{s}_2$.

Our model is not limited to one step ahead predictions: for further predictions, we can simply apply the predicted output $\hat{\mathbf{r}}_j$ as contextual input $\mathbf{r}_j$. During training, instead of always feeding in the ground truth $\mathbf{r}_j$, we sometimes (with 0.5 probability after the 5th epoch) use our predictions $\hat{\mathbf{r}}_j$ as the input for the next step predictions that helps us with being robust to feeding in bad predictions [10].

3.5. Implementation and Training Details

We use a vocabulary V of 30171 words provided by Recipe1M [46]; words are represented by 256-dimensional vectors shared between the sentence encoder and decoder. Our ingredients vocabulary has 3769 ingredients; the one-hot ingredient encodings are mapped into a 1024 dimensional vector $\mathbf{r}_0$. The RNNs are all single-layer LSTMs implemented in PyTorch; LSTM_{se}, LSTM_{ve}, LSTM_{d} have 512 hidden units while LSTM_{d} has 1024. We train our model using the Adam optimizer [26] with a batch size of 50 recipes and a learning rate of 0.001; the text-based model is trained for 50 epochs and the visual encoder for 25 epochs.

4. Tasty Videos Dataset

Our new *Tasty Videos Dataset* has 2511 unique recipes collected from the Buzzfeed website `https://tasty.co`. Each recipe has an ingredient list, step-wise instructions and a video demonstrating the preparation. The recipes feature breakfast, dinner, desserts, and drinks from 185 categories such as cakes, pies, soups. We define a split ratio of 8:1:1 for training, validation and testing, each containing different recipes. Our test setting is therefore zero-shot, as we make predictions on unseen recipes. We further divide the test set into recipes with similarities in the training set, *e.g. "Strawberry Breakfast Muffins"* vs. *"Carrot Cake Muffins"* and those without any similarities *e.g. "Pigs In A Blanket"*.

The Tasty Videos are captured with a fixed overhead camera and focus entirely on preparation of the dish (see Fig. 2). This viewpoint removes the added challenge of distractors and irrelevant actions and while it may not exactly reflect the visual environments one may find in the home, this simplification allows us to focus the scope of our work on modelling the sequential nature of instructional data, which is already a highly challenging and open research topic. The videos are short (on average 1551 frames / 54 seconds) yet contain a challenging number of steps (9 on average). For each recipe step, we annotate the temporal boundaries in which the step occurs within the video, omitting those without visual correspondences, such as alternative recommendations, non-visualized instructions such as *'Preheat oven.'* and stylistic statements such as *'Enjoy!'*.

5. Experiments

5.1. Datasets and Evaluation Measures

We train and evaluate our method with Recipe1M [46], YoucookII [59] and our Tasty Videos. Recipe1M features approximately one million text recipes with a dish name, list of ingredients, and sequence of instructions. YoucookII is a collection of 2000 cooking videos from YouTube from 89 dishes annotated with the temporal boundaries of each step. We use the ingredients and instructions from the Recipe1M training split to learn our sentence encoder, decoder and recipe RNN. To learn the video encoder, we use the aligned instructions and video data from the training split of either YouCookII or Tasty Videos. We evaluate our model's prediction capabilities with text inputs from Recipe1M and video and text inputs from YoucookII and Tasty Videos.

Our predictions are in sentence form; evaluating the quality of generated sentences is known to be difficult in natural language processing [52, 33]. We apply a variety of evaluation measures in order to offer a broad assessment. First, we target the matching of ingredients and verbs, since they indicate the next active objects and actions and are analogous to the assessments made in action anticipation [16]. Second, we evaluate with sentence matching scores BLEU [38] and METEOR [8] which are also used for

	ground truth (GT)	prediction	BLEU1	BLEU4	METEOR	HUMAN1	HUMAN2
ING	bacon, brown sugar, cooking spray, breadsticks						
step1	Preheat oven to 325 degrees F (165 degrees C).	Preheat oven to 400 degrees F.	36.0	0.0	26.0	1.5	1.5
step2	Line 2 baking sheets with aluminum foil or parchment paper and spray with cooking spray.	Line a baking sheet with aluminum foil.	23.0	0.0	23.0	1.0	1.0
step3	Wrap 1 bacon strip around each breadstick, leaving about 1 inch uncovered on each end.	Place bacon strips in a single layer on the prepared baking sheet.	13.0	0.0	9.0	0.5	1.5
step4	Place wrapped breadsticks on the prepared baking sheet.	Place rolls on a baking sheet.	48.0	0.0	30.0	1.5	1.5
step5	Sprinkle brown sugar evenly over breadsticks.	Bake in the preheated oven until breadsticks are golden brown, about 15 minutes.	15.0	0.0	13.0	0.0	1.5
step6	Bake in the preheated oven until bacon is crisp and browned, 50 to 60 minutes.	Bake in preheated oven until bacon is crisp and breadsticks are golden brown, about 15 minutes.	63.0	43.0	36.0	1.0	1.0
step7	Cool breadsticks on a piece of parchment paper or waxed paper sprayed with cooking spray.	Remove from oven and let cool for 5 minutes.	6.0	0.0	4.0	0.5	1.5

Figure 3: Predictions of our text-based method for "Candied Bacon Sticks" along with the automated scores and human ratings. For "HUMAN1" we ask the raters to directly assess how well the predicted steps match the corresponding ground truth (GT) sentences, for "HUMAN2" we ask to judge if the predicted step is still a plausible future prediction, see Sec. 5.7. Our prediction for step 6 matches the GT well while step 5 does not. However, according to "HUMAN2" score, our step 5 prediction is still a plausible future action.

video captioning methods [39, 44, 60]. Note that automated scores are best at indicating precise word matches to ground truth (GT) and often do not match sentences a human would consider equivalent. We therefore conduct a user study and ask people to assess how well the predicted step matches the GT in meaning; if it does not match, we ask if the prediction would be plausible for future steps. This gives flexibility in case predictions do not follow the exact aligned order of the ground truth, *e.g.* due to missing steps not predicted, or steps which are slightly out of order (see Fig. 3)

5.2. Learning of Procedural Knowledge

We first verify the learning of procedural knowledge with a text-only model, evaluating on Recipe1M's test set of 51K recipes. For a recipe of N steps, we predict steps $j+1$ to N, conditioning on steps 1 to j as input context. N varies from recipe to recipe so we separately tally recipes with $N=9$ (4300 recipes; 9 is also the average number of steps in the test set) which we report here. Results over the entire test set follow similar trends and are shown in the Supplementary.

For comparison, we look at the generations from a skip-thought (ST) model [27]. Skip-thought models are trained to decode temporally adjacent sentences from a current encoding, *i.e.* given step j to the encoder, the decoder predicts step $j+1$, and have been shown to be successful in generating continuous text [13, 54, 25]. We train the ST model on the training set of the Recipe1M dataset. Because the ST model generations are not trained to accept an ingredient list as a 0^{th} or initialization step, we make ST predictions only from the second step on-wards.

Key Ingredients: We first compare the recall of ingredients in our predictions to an ST model and a variation of our model trained without ingredients. Rather than directly cross-referencing the ingredient list, we limit the evaluation to ingredients mentioned explicitly in the recipe steps. This is necessary to avoid ambiguities that may arise from specific instructions such as *'add chicken, onion, and bell pepper'* versus the more vague *'add remaining ingredients'*. Furthermore, the ingredient lists in Recipe1M are often automatically generated and may be incomplete. Fig. 5 shows that our model's predictions successfully incorporate relevant ingredients with recall rates as high as 43.3% with the predicted (relative) next step. The overall recall decreases with the (absolute) latter steps, likely due to increased difficulty once the overall number of ingredient occurrences decreases, which tends to happen in later steps.

Compared to the ST model, our predictions' ingredient recall is higher regardless of whether or not ingredients are provided as an initial input. Without ingredient input, the overall recall is lower but after the initial step, our model's recall increases sharply, *i.e.* once it receives some context. We attribute this to the strength of our model to generalize across related recipes, so that it is able to predict relevant co-occurring ingredients. Our predictions include common ingredients such as *salt, butter, eggs* and *water* and also recipe-specific ones such as *couscous, zucchini*, or *chocolate chips*. While the ST model predicts some common ingredients, it fails to predict recipe-specific ingredients.

Key verbs indicate the main action for a step and are also cues for future steps both immediate (*e.g.* after *'adding'* ingredients into a bowl, a common next step may be to *'mix'*) and long-term (*e.g.* after *'preheating'* the oven, one expects to *'bake'*). We tag the verbs in the training recipes with the Natural Language Toolkit [4] and select the 250 most frequent for evaluation. Similar to ingredients, we check for recall of these verbs only if they appear in the ground truth steps. In the ground truth, there are between 1.55 and 1.85 verbs per step, *i.e.* steps often include multiple verbs such as *"add and mix"*. Fig. 4 shows that our model recalls up to 30.9% of the verbs with the predicted next step. Our performance is worst in the first (absolute) steps, due to ambiguities when given only the ingredients without any further knowledge of the recipe. After the first steps, our performance quickly increases and stays consistent across the remaining steps. In comparison, the ST model's best recall is only 20.1% for the next step prediction.

Sentences: Our model is able to predict coherent and plausible instructional sentences as shown in Fig. 3; more predictions can be found in the Supplementary Materials.

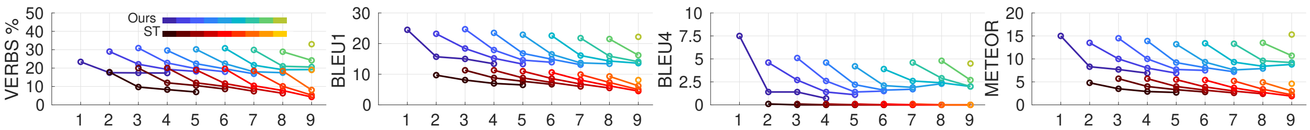

Figure 4: The recall of verbs and sentence scores computed between the predicted and the ground truth sentences for our model (Ours) and the skip-thoughts (ST) model. The x-axes in the plots indicate the step number being predicted in the recipe; each curve begins on the first (relative) prediction, i.e. *the* $(j+1)$*th step after having received steps* 1 *to* j *as input.*

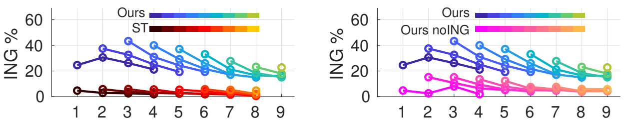

Figure 5: The recall of ingredients predicted by our model (Ours), by our model trained without the ingredients (Ours noING) and the by skip-thoughts model (ST). The x-axes indicate the step number being predicted in the recipe.

We evaluate the entire predicted sentences with BLEU1, BLEU4 and METEOR scores (see Fig 4). For our model, the BLEU1 scores are consistently high, at around 25.0 for the next (relative) step predictions, with a slight decrease towards the end of the recipes. Predictions further than the next step have lower scores, though they stay above 15.0. The BLEU4 scores are highest in the very first step and range between 1.0 and 5.0 over the remaining steps. The high scores at the early steps are because many recipes start with common instructions such as *'Preheat oven to X degrees'* or *'In a large skillet, heat the oil'*. Similarly, we also do well towards the end of recipes, where instructions for serving and garnishing are common, *e.g.*, *'Season with salt and pepper.'*. Trends for the METEOR score are similar.

Our method outperforms the ST model across the board. In fact, predictions up to four steps into the future surpass the ST predictions only one step ahead. This can be attributed to the dedicated long-term modelling of the recipe RNN that allows us to incorporate the context from all sentence inputs up to the present. In contrast, ST are Markovian in nature and can only take the current step into account.

In cooking recipes, one does not only find strict instructional steps, but also suggestions based on experience. An interesting outcome of our model is that it also makes such recommendations. For example, for the ground truth *'If it's too loose place it in the freezer for a little while to freeze.'*, our model predicts *'If you freeze, it will be easier to eat'*.

5.3. Video Predictions

We evaluate our model for making predictions on video inputs on YouCookII's validation set and Tasty Videos' test set. We test two video segmentation settings for inference: one according to ground truth (Ours Visual (GT)) and one based on fixed windows (Ours Visual). In both settings, we sample every fifth frame in these segments and feed their visual features into the recipe RNN as context vectors. Compared to using ground truth segments, the fixed window segments do not have a significant decrease in performance (5%-18%,see Tables 1 and 4 for Tasty and YouCookII respectively). Overall, our method is relatively robust to the window size (see Supplementary) and we report here results for a window of 70 frames for YouCookII and 170 for Tasty.

Through the video encoder, our model can interpret visual evidence and make plausible predictions of next steps (see examples in Fig. 2(b) and 6, more results in Supplementary). Given that the model is first trained on text and then transferred to video, the drop in performance from text to video is as expected, though video results still follow similar trends (see Fig. 7, compare to "Ours Text" in Tables 1 and 4 for Tasty Videos and YouCookII respectively).

We further investigate the influence of the ingredients on the performance of our method. The performance decrease is mainly noticeable in the ingredient scores and the BLUE4 scores. When ingredients are not provided, our method fails to make plausible predictions in the early stages. After the initial steps, our method receives enough context and the scores increase, see Supplementary.

In some instructional scenarios, there may be semi-aligned text that accompanies video, *e.g.* narrations. We test such a setting by training the sentence and video encoder, as well as sentence decoder and recipe RNN jointly for making future step predictions. We concatenate the sentence and video context vectors and then pass them through a linear layer before feeding them as input to the Recipe RNN, and observe that the results are better than our video alone results but not better than our text alone results (see "Ours joint video-text" in Table 1). Even with joint training, it is still difficult to make improvements, which we attribute to the diversity in our videos and variations in the text descriptions for similar visual inputs. On the other hand, when there is accompanying text, our model can be adapted easily and improves the prediction performance.

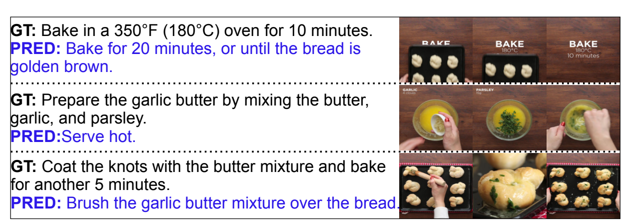

Figure 6: Next step prediction of our visual model: blue sentences are our model's predictions. After baking, our model predicts that the dish should be served, but after visually seeing the butter parsley mixture, it predicts that the knots should be brushed.

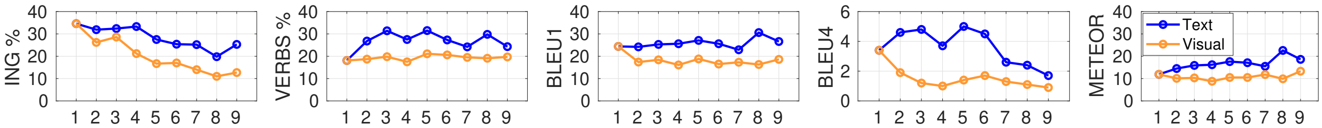

Figure 7: Our results on the Tasty dataset for next step predictions only for our visual and textual model for the recall of predicted ingredients and verbs, and sentence scores. Compared to text, our visual model has a lower performance, but follow similar trends.

Method	ING	VERBS	BLEU1	BLEU4	METEOR
S2VT [53] (GT)	7.59	19.18	18.03	1.10	9.12
S2VT [53], next (GT)	1.54	10.66	9.14	0.26	5.59
End-to-end [60]	-	-	-	0.54	5.48
Ours Visual (GT)	20.40	19.18	19.05	1.48	11.78
Ours Visual	16.66	17.08	17.59	1.23	11.00
Ours Text (100%)	26.09	27.19	26.78	3.30	17.97
Ours Text (50%)	23.01	24.90	25.05	2.42	16.98
Ours Text (25%)	19.43	23.83	23.54	2.03	16.05
Ours Text (0%)	5.80	9.42	10.58	0.24	6.80
Ours Text noING	9.04	22.00	20.11	0.92	13.07
Ours joint video-text	22.27	23.35	21.75	2.33	14.09

Table 1: Evaluations on the Tasty dataset for our visual and text model along with comparison against video captioning [53, 60]. Performance drops when the amount of pre-training decreases. Our method performs better than video captioning.

5.4. Supervised vs. Zero-Shot Learning

We compare the differences of supervised and zero-shot learning on YouCookII. We divide the dataset into four splits based on the 89 dishes and use three splits for training and half of the videos in the fourth split for testing. In the zero-shot setting, the videos from the other half of the fourth split are unused, while in the supervised setting, they are included as part of the training. We report results averaged over the four cross-folds in Table 2.

As expected, the predictions are better when the model is trained under a supervised setting in comparison to zero-shot. This is true for all inputs, with the same drop as observed previously when moving from text to video and when moving from ground truth video segments to fixed window segments. However, the difference between the supervised vs. zero-shot (see Table 2 "Sup. Visual" vs. "Zero Visual") is surprisingly much smaller than the difference between a supervised setting with and without pre-training on Recipe1M ("Sup. Visual" vs. "Sup. Visual no pre-train"). This suggests that having a large corpus for pre-training is more useful than repeated observations for a specific dish.

While the test set of Tasty Videos is fully zero-shot, 183 videos are of recipes which occur with some variations in the training, while 72 are without any variations. As expected, when comparing the predictions on these subsets separately (see Table 3), we observe higher performance on videos with variations, especially for the very difficult BLEU4 score. This suggest that our method generalizes better when it receives visually similar recipes.

5.5. Knowledge Transfer

At the core of our method is the transfer of knowledge from text resources to solve a challenging visual problem. We evaluate the effectiveness of the knowledge transfer by varying the amount of training data from Recipe1M to be used for pre-training. Looking at the averaged scores over all the predicted steps on Tasty Videos, we observe a decrease in all evaluation measures as we limit the amount of data from Recipe1M (see Table 1, "Ours Text" 100%, 50%, etc.), with the most significant decrease occuring for the BLEU4 score. If there is no pre-training, *i.e.* when the model learned only on text from Tasty Videos ("Ours Text (0%)"), the decrease in scores is noticeable for all evaluation criteria. These results again verify that pre-training has a significant effect on our method's performance.

Method	ING	VERBS	BLEU1	BLEU4	METEOR
Sup. Visual (GT)	20.93	24.76	22.11	1.21	10.66
Sup. Visual	18.90	23.15	21.09	1.03	10.22
Sup. Visual no pre-train	2.69	19.43	15.05	0.15	5.89
Sup. Text	24.56	27.24	24.94	1.99	12.50
Zero Visual (GT)	17.77	23.11	20.61	0.84	9.51
Zero Visual	6.04	23.19	20.30	0.76	9.27
Zero Visual no pre-train	1.58	17.83	14.54	0.01	5.03
Zero Text	19.90	24.86	23.06	1.47	10.98

Table 2: Comparison of zero-shot vs. supervised setting (Sup.), on YouCookII [59] by cross validation. Supervised results are better overall. Without pre-training the performance drop is significant.

Method	ING	VERBS	BLEU1	BLEU4	METEOR
w/o variations	14.20	17.08	16.67	0.76	10.00
w/ variations	25.40	20.41	20.54	2.16	13.00
all videos	22.24	19.47	19.45	1.77	12.15

Table 3: Evaluations on the Tasty test set on videos with and without variations in the training set.We do better on variations.

5.6. Comparisons to Video Captioning

We compare our method against different video captioning methods in Tables 1 and 4 for Tasty Videos and YouCookII respectively. Unlike predicting future steps, captioning methods generate sentences after observing their visual data which makes it a much easier task than future prediction. We train and test S2VT [53], an RNN based encoder-decoder approach, on the ground truth segments of the Tasty dataset. Our visual model outperforms this baseline, especially for ingredient recall, by 13%, and with an improvement of 0.3 in BLEU4 score in Table 1. To highlight the difficulty of predicting future steps compared to captioning, we train S2VT [53] for predicting the next step from the observation of the current step (see Table 1 "S2VT [53] next (GT)"). Our visual model outperforms this variation with a big margin for all scores. We also tested the End-to-end Masked Transformer [60] on our dataset and get a BLEU4/METEOR of 0.54 / 5.48 (vs. our 1.23 / 11.00). The poor performance is likely due to the increased dish diversity and difficulty of our dataset vs YouCook2.

We compare our model on the validation set of the

YouCookII dataset against two state-of-the-art video captioning methods [58, 60] in Table 4. End-to-end Masked Transformer [60] performs dense video captioning by both localizing steps and generating descriptions for these steps, while TempoAttn [58] is an RNN-based encoder-decoder approach. Again, even though our task is more difficult than captioning, our method outperforms both of the captioning methods in BLEU4 and METEOR scores. Compared to the state-of-the-art [60], our visual model achieves a METEOR score that is twice as high and a BLEU4 score four times higher. We attribute the better performance of our method compared to the captioning methods to the pre-training on the Recipe1M dataset which allows our model to generalize. Note that for YouCookII, as we use all the videos in the training set, our training is no longer a zero-shot but a supervised scenario.

Method	ING	VERBS	BLEU1	BLEU4	METEOR
TempoAttn(GT) [58]	-	-	-	0.87	8.15
End-to-end(GT) [60]	-	-	-	1.42	11.20
Ours Visual (GT)	21.36	27.55	23.71	1.66	11.54
TempoAttn [58]	-	-	-	0.08	4.62
End-to-end [60]	-	-	-	0.30	6.58
Ours Visual	17.64	25.11	22.55	1.38	10.71
Ours Text	24.60	29.39	26.49	2.66	13.31

Table 4: Comparison against captioning methods on the YouCookII [59] validation set. We perform better than the state-of-the-art captioning methods.

5.7. Human Ratings

We ask human raters to directly assess how well the predicted steps match the ground truth with scores 0 ('not at all'), 1 ('somewhat') or 2 ('very well'). If the prediction receives a score of 0, we additionally ask the human to judge if the predicted step is still a plausible future prediction, again with the same scores of 0 ('not at all'), 1 ('somewhat'), or 2 ('very likely'). We conduct this study with 3 people on a subset of 30 recipes from the test set, each with 7 steps, and present their ratings in Fig. 8 while comparing them to automated sentence scores.

In Fig. 8, the upper graph shows the results of the human raters and the lower graph shows the automated sentence scores. Raters report a score close to 1 for the initial step predictions indicating that our method, even by only seeing the ingredients, can start predicting plausible steps. Scores increase towards the end of the recipe and are lowest at step 3. The average score of the predicted steps being a possible future prediction are consistently high across all steps. Even if the predicted step does not exactly match the ground truth, human raters still consider it possible for the future, including the previously low rating for step 3. Overall, the ratings indicate that the predicted steps are plausible.

The lower graph in Fig. 8 shows automated scores for the same user study recipes. The left plot shows the standard scores for the predicted sentences matching the ground truth; overall, trends are very similar to the user study, including the low-scoring step 3. To match the second setting

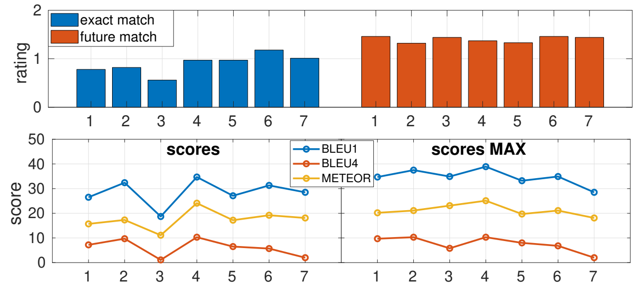

Figure 8: Comparison of human ratings (upper graph) versus automated sentence scores (lower graph) over a subset of 30 recipes.

of the user study, we compute the sentence scores between the predicted sentence $\hat{s}_j$ and all future ground truth steps $\{s_j, s_{j+1}, s_{j+2}, s_{j+3}\}$ and select the step with the maximum score as our future match. These scores are plotted in the lower right of Fig. 8; similar to the human study, sentence scores increase overall.

5.8. Ablation Study

Since our method is modular, we conduct an ablation study to check the interchangeability of the sentence encoder on the Recipe1M dataset [46]. Instead of using our own sentence encoder, we represent the sentences using ST vectors trained on the Recipe1M dataset, as provided by the authors [46]. These vectors have been shown to perform well for their recipe retrieval. Our results, presented in the supplementary text show that our sentence encoder performs on par with ST encodings. However, our encoder, model and decoder can all be trained jointly and do not require a separate pre-training of a sentence autoencoder.

6. Conclusion

In this paper we present a method for zero-shot action anticipation in videos. Our model learns to generalize instructional knowledge from the text domain. Applying this knowledge to videos allows us to tackle the challenging task of predicting steps of complex tasks from visual data, which is otherwise ruled out because of scarcity of or difficulty to annotate training data. We present a new, diverse dataset of cooking videos, which is of high interest for the community. We successfully validate our method's performance on both text and video data. We show that our model is able to produce coherent and plausible future steps. We conclude that our knowledge transfer strategy works much better than captioning methods and generalizes well on different datasets. In the future we hope to include more information into our model, such as the title of the recipe.

Acknowledgments This work has been partly funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) YA 447/2-1 and GA 1927/4-1 (FOR 2535 Anticipating Human Behavior) and partly by the Singapore Ministry of Education Academic Research Fund Tier 1. We thank Sven Behnke and Juergen Gall for useful discussions.

References

[1] Instructables. `https://www.instructables.com/`.

[2] Wikihow-how to do anything. `http://www.wikihow.com/`.

[3] Moley - The World's First Robotic Kitchen. `http://www.moley.com/`, 2018.

[4] Natural Language Toolkit: NLTK 3.3 documentation. `http://www.nltk.org/`, 2018.

[5] Yazan Abu Farha, Alexander Richard, and Juergen Gall. When will you do what? Anticipating temporal occurrences of activities. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2018*.

[6] Jean-Baptiste Alayrac, Piotr Bojanowski, Nishant Agrawal, Josef Sivic, Ivan Laptev, and Simon Lacoste-Julien. Unsupervised learning from narrated instruction videos. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2016*.

[7] Jimmy Ba, Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.

[8] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *ACL Workshop, 2005*.

[9] M Beetz, U Klank, A Maldonado, D Pangercic, and T Ruhr. Robotic roomates making pancakes - Look into perception-manipulation loop. In *International Conference on Robotics and Automation (ICRA), Workshop on Mobile Manipulation: Integrating Perception and Manipulation, 2011*.

[10] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems (NIPS), 2015*.

[11] Piotr Bojanowski, Rémi Lajugie, Francis Bach, Ivan Laptev, Jean Ponce, Cordelia Schmid, and Josef Sivic. Weakly supervised action labeling in videos under ordering constraints. In *European Conference on Computer Vision (ECCV), 2014*.

[12] Micael Carvalho, Rémi Cadène, David Picard, Laure Soulier, Nicolas Thome, and Matthieu Cord. Cross-modal retrieval in the cooking context: Learning semantic text-image embeddings. In *ACM SIGIR Conference on Research & Development in Information Retrieval, 2018*.

[13] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014*.

[14] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *Conference on Empirical Methods in Natural Language Processing (EMNLP), 2017*.

[15] Robert Dale. Cooking up referring expressions. In *ACL*, 1989.

[16] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The epic-kitchens dataset. In *European Conference on Computer Vision (ECCV), 2018*.

[17] Pradipto. Das, Chenliang. Xu, Richard F. Doell, and Jason J. Corso. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2013*.

[18] Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning. In *Conference on Robot Learning, 2017*.

[19] Kristian J. Hammond. Chef: A model of case-based planning. In *AAAI*, 1986.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2016*.

[21] Luis Herranz, Weiqing Min, and Shuqiang Jiang. Food recognition and recipe analysis: integrating visual content, context and external knowledge. *ArXiv*, abs/1801.07239, 2018.

[22] Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning distributed representations of sentences from unlabelled data. In *Proceedings of NAACL-HLT, 2016*.

[23] Minh Hoai and Fernando De la Torre. Max-margin early event detectors. *International Journal of Computer Vision*, 107:191–202, 2014.

[24] De-An Huang, Li Fei-Fei, and Juan Carlos Niebles. Connectionist temporal modeling for weakly supervised action labeling. In *European Conference on Computer Vision (ECCV), 2016*.

[25] Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. Globally coherent text generation with neural checklist models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP), 2016*.

[26] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference for Learning Representations (ICLR), 2015*.

[27] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *Advances in Neural Information Processing Systems (NIPS), 2015*.

[28] Hema Koppula and Ashutosh Saxena. Anticipating human activities using object affordances for reactive robotic response. *Transactions on pattern analysis and machine intelligence, (PAMI)*, 38:14–29, 2016.

[29] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2014*.

[30] H. Kuehne, A. B. Arslan, and T. Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2014*.

[31] Tian Lan, Tsung-Chuan Chen, and Silvio Savarese. A hierarchical representation for future action prediction. In *European Conference on Computer Vision (ECCV), 2014*.

[32] Dahua Lin, Chen Kong, Sanja Fidler, and Raquel Urtasun. Generating multi-sentence lingual descriptions of indoor scenes. *British Machine Vision Conference (BMVC), 2015*.

[33] Adam Lopez. Statistical machine translation. *ACM Computing Surveys (CSUR)*, 2008.

[34] Jonathan Malmaud, Jonathan Huang, Vivek Rathod, Nicholas Johnston, Andrew Rabinovich, and Kevin Murphy. Whats cookin? interpreting cooking videos using text, speech and vision. In *North American Chapter of the Association for Computational Linguistics (NAACL), 2015*.

[35] Weiqing Min, Shuqiang Jiang, Shuhui Wang, Jitao Sang, and Shuhuan Mei. A delicious recipe analysis framework for exploring multi-modal recipes with various attributes. In *ACM International Conference on Multimedia, 2017*.

[36] Shinsuke Mori, Hirokuni Maeta, Tetsuro Sasada, Koichiro Yoshino, Atsushi Hashimoto, Takuya Funatomi, and Yoko Yamakata. Flowgraph2text: Automatic sentence skeleton compilation for procedural text generation. In *INLG, 2014*.

[37] Richard G. Morris, Scott H. Burton, Paul Bodily, and Dan Ventura. Soup over bean of pure joy: Culinary ruminations of an artificial chef. In *ICCC, 2012*.

[38] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics, 2002*.

[39] Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. Grounding action descriptions in videos. *Trans. Ass. Computational Linguistics*, 2013.

[40] Alexander Richard and Juergen Gall. Temporal action detection using a statistical language model. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2016*.

[41] Alexander Richard, Hilde Kuehne, and Juergen Gall. Weakly supervised action learning with rnn based fine-to-coarse modeling. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2017*.

[42] Anna Rohrbach, Marcus Rohrbach, Wei Qiu, Annemarie Friedrich, Manfred Pinkal, and Bernt Schiele. Coherent multi-sentence video description with variable level of detail. In *German Conference on Pattern Recognition (GCPR), 2014*.

[43] Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. A database for fine grained activity detection of cooking activities. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2012*.

[44] Marcus Rohrbach, Wei Qiu, Ivan Titov, Stefan Thater, Manfred Pinkal, and Bernt Schiele. Translating video content to natural language descriptions. In *International Conference on Computer Vision (ICCV), 2013*.

[45] Michael S Ryoo. Human activity prediction: Early recognition of ongoing activities from streaming videos. In *International Conference on Computer Vision (ICCV), 2011*.

[46] Amaia Salvador, Nicholas Hynes, Yusuf Aytar, Javier Marin, Ferda Ofli, Ingmar Weber, and Antonio Torralba. Learning cross-modal embeddings for cooking recipes and food images. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2017*.

[47] Fadime Sener and Angela Yao. Unsupervised learning and segmentation of complex activities from video. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2018*.

[48] Ozan Sener, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Unsupervised semantic parsing of video collections. In *International Conference on Computer Vision (ICCV), 2015*.

[49] Niko Sünderhauf, Oliver Brock, Walter J. Scheirer, Raia Hadsell, Dieter Fox, Jürgen Leitner, Ben Upcroft, Pieter Abbeel, Wolfram Burgard, Michael Milford, and Peter I. Corke. The limits and potentials of deep learning for robotics. *I. J. Robotics Res.*, 37:405–420, 2018.

[50] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NIPS), 2014*.

[51] Moritz Tenorth, Alexander Clifford Perzylo, Reinhard Lafrenz, and Michael Beetz. Representation and Exchange of Knowledge about Actions, Objects, and Environments in the RoboEarth Framework. *Transactions on Automation Science and Engineering (T-ASE)*, 10:643–651, 2013.

[52] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2015*.

[53] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence-video to text. In *International Conference on Computer Vision (ICCV), 2015*.

[54] Oriol Vinyals and Quoc Le. A neural conversational model. *ICML Deep Learning Workshop*, 2015.

[55] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Anticipating visual representations from unlabeled video. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2016*.

[56] Chenxia Wu, Jiemi Zhang, Bart Selman, Silvio Savarese, and Ashutosh Saxena. Watch-bot: Unsupervised learning for reminding humans of forgotten actions. In *International Conference on Robotics and Automation (ICRA), 2016*.

[57] Zhen Xu, Laiyun Qing, and Jun Miao. Activity autocompletion: Predicting human activities from partial videos. In *International Conference on Computer Vision (ICCV), 2015*.

[58] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure. In *International Conference on Computer Vision (ICCV), 2015*.

[59] Luowei Zhou, Chenliang Xu, and Jason J Corso. Towards automatic learning of procedures from web instructional videos. In *AAAI Conference on Artificial Intelligence, 2018*.

[60] Luowei Zhou, Yingbo Zhou, Jason J Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In *Conference on Computer Vision and Pattern Recognition (CVPR), 2018*.

[61] Yipin Zhou and Tamara L Berg. Temporal perception and prediction in ego-centric video. In *International Conference on Computer Vision (ICCV), 2015*.