# Visual Goal-Step Inference using wikiHow

**Yue Yang, Artemis Panagopoulou, Qing Lyu, Li Zhang,**
**Mark Yatskar, Chris Callison-Burch**
Department of Computer and Information Science, University of Pennsylvania
{yueyang1,artemisp,lyuqing,zharry,myatskar,ccb}@seas.upenn.edu

## Abstract

Understanding what sequence of steps are needed to complete a goal can help artificial intelligence systems reason about human activities. Past work in NLP has examined the task of goal-step inference for text. We introduce the visual analogue. We propose the Visual Goal-Step Inference (VGSI) task, where a model is given a textual goal and must choose which of four images represents a plausible step towards that goal. With a new dataset harvested from wikiHow consisting of 772,277 images representing human actions, we show that our task is challenging for state-of-the-art multimodal models. Moreover, the multimodal representation learned from our data can be effectively transferred to other datasets like HowTo100m, increasing the VGSI accuracy by 15 - 20%. Our task will facilitate multimodal reasoning about procedural events.

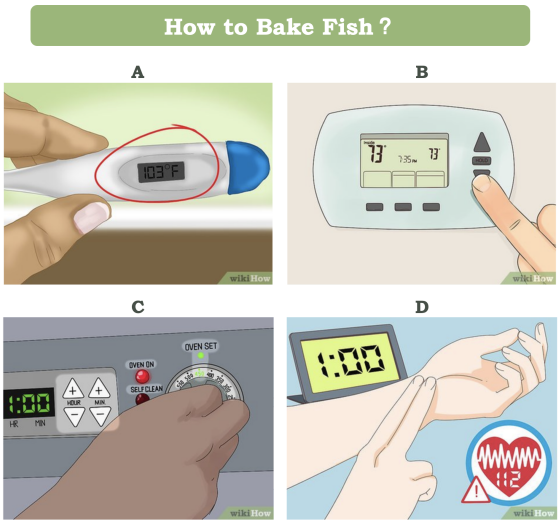



Figure 1: An example Visual Goal-Step Inference Task: given a text goal (*bake fish*), select the image (C) that represents a step towards that goal.

## 1 Introduction

Recently, there has been growing attention on the representation of complex events, with renewed interest in script learning and commonsense reasoning (Park and Motahari Nezhad, 2018; Mujtaba and Mahapatra, 2019; Li et al., 2020). One aspect of event representation is the relationship between high-level goals and the steps involved (Zhang et al., 2020b,a). For example, given a goal (e.g. "change a tire"), an intelligent system should be able to infer what steps need to be taken to accomplish the goal (e.g. "place the jack under the car", "raise the jack"). In most work, events are represented as text (Zellers et al., 2018; Coucke et al., 2018; Zhang et al., 2019), while they could have different modalities in the real world.

Learning *goal-step relations* in a multimodal fashion is an interesting challenge since it requires reasoning beyond image captioning. We contend that multimodal event representation learning will have interesting implications for tasks such as schema learning (Li et al., 2020, 2021) to mitigate reporting bias (Gordon and Van Durme, 2013) since steps are often not explicitly mentioned in a body of text. For instance, if a robot is asked to "get a slice of cake," it has to know that it should "take the cake out of the box", "cut a slice", "put it on a plate", and then "take the plate to the user". Such steps are commonsense to people and thus rarely specified explicitly, making them hard to infer from textual data. However, with multimodal learning, we could obtain such details from visual signals. This multimodal goal-step relation could also be used for vision-enabled dialog systems[1] to recognize what task a user is completing and provide helpful recommendations.

We propose a new task called **Visual Goal-Step Inference (VGSI)**: given a textual goal and multiple images representing candidate events, a model must choose one image which constitutes a reasonable step towards the given goal. This means that a

[1]Like the Alexa Prize Taskbot Challenge.



*Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2167–2179
November 7–11, 2021. 

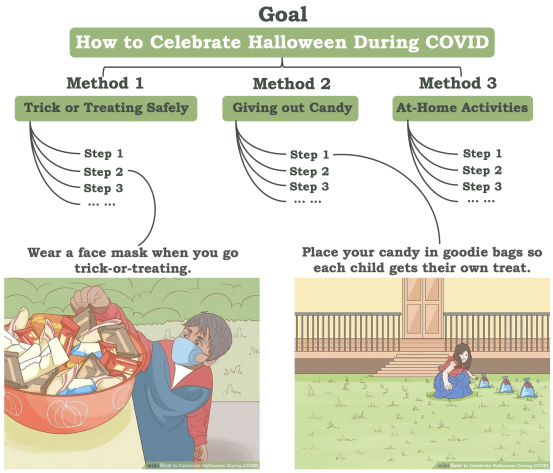



Figure 2: Hierarchical multimodality of wikiHow.

model should correctly recognize not only the specific action illustrated in an image (e.g., "turning on the oven", in Figure 1), but also the intent of the action ("baking fish").

We collect data from **wikiHow** articles, where most steps are illustrated with images. Our VGSI training set is constructed using three sampling strategies to select negative image candidates as distractors. In the format of multiple-choice and image retrieval, we evaluate four state-of-the-art multimodal models: DeViSE (Frome et al., 2013), Similarity Networks (Wang et al., 2018), Triplet Networks (Hoffer and Ailon, 2015), and LXMERT (Tan and Bansal, 2019) to human performance. It is observed that SOTA models designed for caption-based multimodal tasks (Karpathy et al., 2014; Johnson et al., 2016) struggle on VGSI, exhibiting a 40% gap in accuracy from human performance when using a challenging sampling strategy.

One limitation of wikiHow is that most images are drawings rather than photos (which are more typically used in computer vision research). The knowledge learned from wikiHow is nevertheless useful when applied to real photos. We demonstrate this by pre-training a triplet-network on our wikiHow VGSI task and then conducting transfer learning on out-of-domain datasets. Our experiments show that pre-trained models can effectively transfer the goal-step knowledge to task-oriented video datasets, such as COIN (Tang et al., 2019) and Howto100m (Miech et al., 2019). In addition, we design an aggregation model on top of SOTA models which treats wikiHow as a knowledge base that further increases the transfer learning performance (see Appendix C).

We make three key contributions: (1) We propose the VGSI task, which is more challenging than traditional caption-based image-text matching tasks and requires the model to have an intermediate reasoning process about goal-step relations. (2) To study the VGSI task, we collect a multimodal dataset from wikiHow which contains over 770k images. (3) Through transfer learning, we show that the knowledge learned from our dataset can be readily applied to out-of-domain datasets, with an accuracy improvement of 15-20% on VGSI.

| Category | Goals | Methods | Steps | Images |
|---|---|---|---|---|
| Health | 7.8k | 19.1k | 97.5k | 111.8k |
| Home and Garden | 5.9k | 16.0k | 82.9k | 85.4k |
| Education & Communications | 4.7k | 12.4k | 61.2k | 66.1k |
| Food & Entertaining | 4.6k | 11.6k | 62.0k | 69.0k |
| Finance & Business | 4.4k | 11.8k | 59.3k | 66.8k |
| Pets & Animals | 3.5k | 9.5k | 45.3k | 48.0k |
| Personal Care & Style | 3.4k | 9.0k | 46.1k | 48.9k |
| Hobbies & Crafts | 2.8k | 7.5k | 40.9k | 42.7k |
| Computers & Electronics | 2.6k | 6.1k | 31.5k | 36.2k |
| Arts & Entertainment | 2.5k | 6.8k | 35.4k | 37.2k |
| Total | 53.2k | 155.3k | 772.3k | 772.3k |

Table 1: Number of goals, methods, steps and images in the top 10 wikiHow categories.

## 2 wikiHow as Multimodal Resource

We use wikiHow as the data source for VGSI because it has been successfully adopted in prior work for procedural learning (Zhou et al., 2019) and intent detection (Zhang et al., 2020a) in the language domain. As shown in Figure 2, each wikiHow article contains a high-level *goal* and one or more different *methods*[2] to achieve it. Each method then includes a series of specific *steps*, typically accompanied with corresponding images.

The format of wikiHow articles provides a hierarchical multimodal relationship between images and sentences. We can obtain three types of text-image pairs from wikiHow, in increasing specificity: goal-image, method-image, and step-image. However, these text-image pairs are not enough information for a system to succeed on VGSI; it also needs the appropriate background knowledge. For the example in Figure 2, the system needs to know that "Trick-or-Treating" and "candies" are Halloween traditions and that a "mask" is required during "COVID-19".

In total, as shown in Table 1, the corpus consists of 53,189 wikiHow articles across various categories of everyday tasks, 155,265 methods, and 772,294 steps with corresponding images [3]

[2]In some articles, they use *parts* instead of *methods*.

[3]Both datasets and code are available here.

## 3 Methods

### 3.1 Problem Formulation

Given a high-level goal $G$—defined as a sequence of words—and an image $I \in \mathbb{R}^{3\times h\times w}$—with the dimension of 3 color channels, the width, and the height—the model outputs the matching score:

$$match(G, I) = F(X_G, X_I) \quad (1)$$

in which, $X_G \in \mathbb{R}^{d_G}$ and $X_I \in \mathbb{R}^{d_I}$ are the feature representations of the goal and the image, respectively. $F$ is the scoring function that models the interactions between the two representations. At inference time, the model will choose the candidate with the highest matching score as the prediction.

### 3.2 Models

**DeViSE** takes in the pre-trained embedding vectors from the two modalities and maps the source vector onto the span of the target vector. DeViSE is trained only on the positive pairs $(G, I)$ and projects an image embedding onto the same dimension as the goal with L2 normalization. Then it computes the cosine similarity of the two normalized vectors as the matching score.

**Similarity Network** Each branch of the similarity network maps one modality to the joint span and executes point-wise multiplication to construct a joint vector. The last layer is fully-connected with softmax activation and outputs an array of size two to denote the weights of each class for binary classification. We compute the matching score by taking the dot product $[1, -1]$ with the output.

**Triplet Network** requires the input to be the format of a triplet $(G, I_{pos}, I_{neg})$. Three branches in the network map the three embeddings to the same joint span, such that the branches of positive and negative images share the same weight. The network learns the cross-modality by minimizing the positive pair distance and maximizing the negative pair distance. We choose cosine distance as the metric which is also used as the matching score.

**LXMERT** is a multimodal encoder that aims to ground text to images through attention layers. The image input is represented as a sequence of objects and the sentence input is a sequence of words. LXMERT utilizes two single-modality transformer encoders (language and object encoders) and a cross-modality transformer encoder to achieve the attention mechanism and capture the relationship between the two modalities. Same as the similarity network, LXMERT is trained as a binary classifier.

| Model | Sampling Strategy (Test Size) | | |
|---|---|---|---|
| | Random (153,961) | Similarity (153,770) | Category (153,961) |
| Random | .2500 | .2500 | .2500 |
| DeViSE | .6719 | .3364 | .4558 |
| Similarity Net | .6895 | .6226 | .4983 |
| LXMERT | .7175 | .4259 | .2886 |
| Triplet Net (GloVe) | .7251 | .7450 | .5307 |
| Triplet Net (BERT) | **.7280** | **.7494** | **.5360** |
| Human | .8450 | .8214 | .7550 |

Table 2: Accuracy of SOTA models on the wikiHow VGSI test set with different sampling strategies (sample size is shown in parentheses).

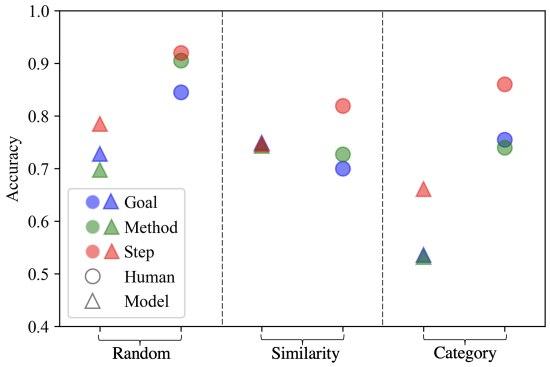



Figure 3: Accuracy of human (circles) and model (triangles) on the modified wikiHow VGSI test set with different textual input (e.g., in Fig 1, the *goal* prompt will be replaced by *method* - "Baking the Fish." or *step* - "Preheat the oven.").

## 4 Experimental Setup

### 4.1 Multiple-Choice Sampling

We formulate the task as a 4-way multiple choice question, which is easy for evaluating the image-text matching performance and is feasible for human annotation. Specifically, a model is given a textual goal & four images to predict the most reasonable step towards the goal. We utilize three sampling strategies to obtain negative candidates:

**Random Strategy** We randomly pick three different articles and select one image by chance from each article as the negative sample.

**Similarity Strategy** We greedily select the most similar images based on the feature vectors and use FAISS (Johnson et al., 2019) to retrieve the top-3 most similar images from three different articles.

**Category Strategy** The three negative samples are randomly selected from articles within the same wikiHow category as the prompt goal.

In addition to the multiple-choice format, we also evaluate VGSI in a more realistic goal-image retrieval format (see Appendix B).

### 4.2 Human Annotation

Considering that VGSI is a novel task, we also evaluate how difficult it is for humans. All of our six human annotators are graduate students with good English proficiency. For each annotation test, we selected 100 samples from the testing set. A pair of annotators completed each test and their scores were averaged.

### 4.3 Evaluation Metrics

We report both model and human accuracy for the multiple-choice task. For the retrieval task, we adopt recall at $k$ (recall@$k$) and median rank (Med r) to measure the performance (see Appendix B).

## 5 Results

### 5.1 In-Domain Results

Table 2 shows the performance of the models and humans on the wikiHow dataset. The Triplet Network with BERT embeddings has the best performance. However, there is still a big gap between human and model performance, indicating that VGSI is challenging for even SOTA models. LXMERT performs badly using similarity and category strategies presumably because it heavily depends on grounding objects, and negative samples generated by these two strategies could share similar objects but refer to different goals. Figure 3 demonstrates that both humans and models perform better with lower-level texts as prompt, which reflects that our VGSI task is more challenging.

### 5.2 Transfer Learning

To robustly show the potential of wikiHow as a multimodal transfer learning resource, we compare it with two existing caption-based datasets, Flickr30K (Plummer et al., 2015) and MSCOCO (Vinyals et al., 2016), which are used as pre-training alternatives. We use the official train/val split for each dataset and pre-train two models separately on Flickr and MSCOCO using the same multiple-choice sampling strategies as VGSI.

#### 5.2.1 Target Datasets & Keyframe Extraction

Our transfer targets include COIN and Howto100m, both large-scale datasets of instructional videos. Each video depicts the process of accomplishing a high-level goal, mostly everyday tasks. Since these two datasets are video-based while our task is image-based, we apply a key frame extraction heuristic to get critical frames from videos. We then consider the key frames as steps, thus converting the datasets into the VGSI format.

| | | Sampling Strategy | | |
|---|---|---|---|---|
| **PT-Data** | **FT?** | **Random** | **Similarity** | **Category** |
| - | ✓ | .6649 | .5085 | .5216 |
| Flickr30K | ✗ | .4903 | .5103 | .3919 |
| | ✓ | .7006 | .5823 | .5495 |
| MSCOCO | ✗ | .5349 | .5401 | .4071 |
| | ✓ | .7481 | .6180 | .5536 |
| Howto100m | ✗ | .5694 | .5811 | .3989 |
| | ✓ | .6948 | .6104 | .5436 |
| wikiHow | ✗ | **.6245** | **.6309** | **.4586** |
| | ✓ | **.7639** | **.6854** | **.5659** |
| Human | - | .9695 | .8500 | .8682 |

Table 3: Transfer performance (4-way multiple choice accuracy) on COIN. PT stands for pre-training, FT for fine-tuning. FT results are obtained by fine-tuning the model on 5 examples of the COIN training set (i.e., 5-shot). **Red** numbers indicate the best zero-shot performance. **Blue** numbers are the best fine-tuned results.

| | | Sampling Strategy | | |
|---|---|---|---|---|
| **PT-Data** | **FT?** | **Random** | **Similarity** | **Category** |
| - | ✓ | .6005 | .6096 | .4434 |
| Flickr30K | ✗ | .4837 | .5398 | .3856 |
| | ✓ | .6207 | .6408 | .4740 |
| MSCOCO | ✗ | .5099 | .5715 | .3958 |
| | ✓ | .6340 | .6640 | .4794 |
| COIN | ✗ | .5067 | .5161 | .3978 |
| | ✓ | .6170 | .6343 | .4638 |
| wikiHow | ✗ | **.6556** | **.6754** | **.4750** |
| | ✓ | **.6855** | **.7249** | **.5143** |
| Human | - | .8300 | .7858 | .7550 |

Table 4: Transfer performance (4-way multiple choice accuracy) on Howto100m. FT results are obtained by fine-tuning the model on the full training set.

**Howto100m:** We randomly select 1,000 goals and one video for each goal. To extract key frames, we apply k-means clustering in the feature space of the frames of each video and select the closest frame to each cluster center. We further filter these frames by manually removing unrelated frames such as the introduction, transition animations, repetitive frames, etc. We finally obtain 869 goals[4] with 24.7 frames for each goal.

**COIN:** We randomly select 900 videos (5 videos per goal) to construct the test set, and use the remaining 9,709 videos for training. Since COIN has annotations of textual steps and their corresponding video segment, we randomly select one frame within each video segment as a VGSI candidate, resulting in 230.1 frames per goal.

[4]Some goals have no valid frames remaining after the annotation, and are therefore removed altogether.

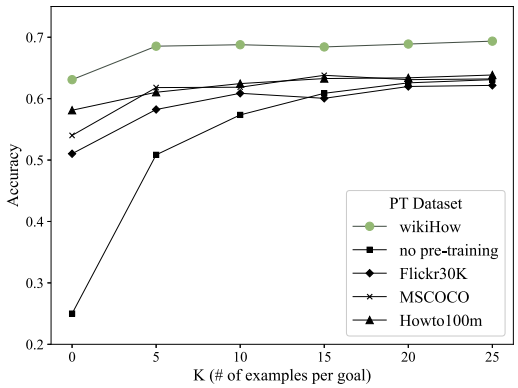



Figure 4: Few-shot performance on COIN (similarity sampling) with different pre-training datasets vs. the number of examples per goal.

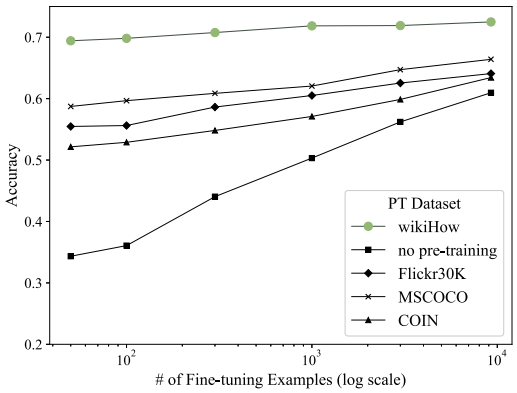



Figure 5: Transfer performance on Howto100m (similarity sampling) with different pre-training datasets vs. the number of training examples.

Then we use these frames to construct the multiple-choice examples with the same three sampling strategies. We also compare using wikiHow against using COIN and Howto100m as pre-training data to perform transfer learning to each other since both are instructional video datasets.

#### 5.2.2 Transfer Learning Performance

We use two different transfer learning setups for COIN[5] and Howto100m. For COIN, we formulate the test as a $K$-shot learning task where $K$ is the number of VGSI training examples for each goal. The 180 goals for testing are seen during training to simulate the scenario where we have some instances of a task. For Howto100m, we split the 869 goals into 8:2 for training and testing, where the test goals are unseen during training.

Tables 3 and 4 both indicate that pre-training on wikiHow can improve VGSI performance on out-of-domain datasets. Especially for the Howto100m results, the model pre-trained on wikiHow without fine-tuning outperforms even those pre-trained on other caption-based datasets that were fine-tuned on wikiHow. This is strong evidence that wikiHow can serve as a useful knowledge resource since the learned multimodal representation can be directly applied to other datasets.

To further validate whether the advantages of pre-training on wikiHow persist with the increasing number of fine-tuning examples, we report the performance with $K \in \{0, 5, 10, 15, 20, 25\}$ for COIN and training examples ranging from 50 to 9,249 (full) for Howto100m. Shown in Figure 4 & 5, the model pre-trained on wikiHow consistently outperforms those pre-trained on the other datasets by significant margins with the increase of fine-tuning examples. The curve of wikiHow does not converge with the other curves even with the maximum number of training examples, which reflects that wikiHow could be a reliable pre-training data source for both low- and rich-resource scenarios.

[5]The small number of goals in COIN leads to an extreme imbalance between video frames and texts, which makes it hard for training. Thus there is no train/test split on goals.

## 6 Conclusion

In this paper, we propose the novel Visual Goal-Step Inference task (VGSI), a multimodal challenge for reasoning over procedural events. We construct a dataset from wikiHow and show that SOTA multimodal models struggle on it. Based on the transfer learning results on Howto100m and COIN, we validate that the knowledge harvested from our dataset could transfer to other domains. The multimodal representation learned from VGSI has strong potential to be useful for NLP applications such as multimodal dialog systems.

## Acknowledgments

This research is based upon work supported in part by the DARPA KAIROS Program (contract FA8750-19-2-1004), the DARPA LwLL Program (contract FA8750-19-2-0201), and the IARPA BETTER Program (contract 2019-19051600004). Approved for Public Release, Distribution Unlimited. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, IARPA, or the U.S. Government.

We thank Chenyu Liu for annotations. We also thank Simmi Mourya, Keren Fuentes, Carl Vondrick, Zsolt Kira, Mohit Bansal, Lara Martin, and anonymous reviewers for their valuable feedback.

## References


Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. 2013. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*, pages 2121–2129.

Jonathan Gordon and Benjamin Van Durme. 2013. Reporting bias and knowledge acquisition. In *Proceedings of the 2013 workshop on Automated knowledge base construction*, pages 25–30.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Elad Hoffer and Nir Ailon. 2015. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, pages 84–92. Springer.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*.

Justin Johnson, Andrej Karpathy, and Li Fei-Fei. 2016. Densecap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4565–4574.

Andrej Karpathy, Armand Joulin, and Li Fei-Fei. 2014. Deep fragment embeddings for bidirectional image sentence mapping. *arXiv preprint arXiv:1406.5679*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Manling Li, Sha Li, Zhenhailong Wang, Lifu Huang, Kyunghyun Cho, Heng Ji, Jiawei Han, and Clare Voss. 2021. Future is not one-dimensional: Graph modeling based complex event schema induction for event prediction. *arXiv preprint arXiv:2104.06344*.

Manling Li, Qi Zeng, Ying Lin, Kyunghyun Cho, Heng Ji, Jonathan May, Nathanael Chambers, and Clare Voss. 2020. Connecting the dots: Event graph schema induction with path language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 684–695.

Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. 2019. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *Proceedings of the IEEE international conference on computer vision*, pages 2630–2640.

Dena Mujtaba and Nihar Mahapatra. 2019. Recent trends in natural language understanding for procedural knowledge. In *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 420–424. IEEE.

Hogun Park and Hamid Reza Motahari Nezhad. 2018. Learning procedures from text: Codifying how-to procedures in deep neural networks. In *Companion Proceedings of the The Web Conference 2018*, pages 351–358.

Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2015. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2015. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and Jie Zhou. 2019. Coin: A large-scale dataset for comprehensive instructional video analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2016. Show and tell: Lessons learned from the 2015 mscoco image captioning challenge. *IEEE transactions on pattern analysis and machine intelligence*, 39(4):652–663.

Liwei Wang, Yin Li, Jing Huang, and Svetlana Lazebnik. 2018. Learning two-branch neural networks for image-text matching tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):394–407.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and Philip Yu. 2019. Joint slot filling and intent detection via capsule neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5259–5267, Florence, Italy. Association for Computational Linguistics.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020a. Intent detection with WikiHow. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 328–333, Suzhou, China. Association for Computational Linguistics.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020b. Reasoning about goals, steps, and temporal ordering with WikiHow. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4630–4639, Online. Association for Computational Linguistics.

Yilun Zhou, Julie Shah, and Steven Schockaert. 2019. Learning household task knowledge from WikiHow descriptions. In *Proceedings of the 5th Workshop on Semantic Deep Learning (SemDeep-5)*, pages 50–56, Macau, China. Association for Computational Linguistics.

# A Model Implementation Details

## A.1 Architecture and Loss Function

### A.1.1 DeViSE

The Deep-Visual Semantic Embedding (DeViSE) model takes in the pre-trained embedding vectors from two modalities and maps the source vector onto the span of the target vector representation. First, the DeViSE model is only trained on the related (positive) pairs $(G, I)$, and we map the image to the goal ($I \rightarrow G$). Then, the model projects the image embedding onto the same dimension as the goal and we apply L2 normalization to obtain the unit vectors:

$$\begin{aligned} \hat{X}_I &= L_2N(X_I W_{I\rightarrow G}) \\ \hat{X}_G &= L_2N(X_G) \end{aligned} \tag{2}$$

where, $L_2N$ stands for L2 normalization and $W_{I\rightarrow G} \in \mathbb{R}^{d_I \times d_G}$ is the weight.

Then the DeViSE model uses a similarity function (here we choose cosine distance) to compute the distance between $\hat{X}_I$ and $\hat{X}_G$ as the loss:

$$\begin{aligned} \mathcal{L}_{DeViSE} &= cos(\hat{X}_I, \hat{X}_G) \\ match(G, I)_{DeViSE} &= 1 - cos(\hat{X}_I, \hat{X}_G) \end{aligned} \tag{3}$$

In which $cos$ means the cosine distance. For DeViSE, the matching score is the cosine similarity between the two unit vectors.

### A.1.2 Similarity Network

A Similarity Network is one type of two-branch networks for matching an image and text. It is a supervised model which takes in $(G_i, I_i, y_i)$, and $y_i \in \{0, 1\}$ is the binary label that indicates whether $G_i$ and $I_i$ are related or not.

Each branch of the network maps one modality to the cross-modality span and executes pointwise multiplication to construct a joint vector:

$$\begin{aligned} \hat{X}_I &= L_2N(X_I W_{I\rightarrow J}) \\ \hat{X}_G &= L_2N(X_G W_{G\rightarrow J}) \\ X_J &= \hat{X}_I \odot \hat{X}_G \end{aligned} \tag{4}$$

in which, $W_{I\rightarrow J} \in \mathbb{R}^{d_I \times d_J}$ and $W_{G\rightarrow J} \in \mathbb{R}^{d_G \times d_J}$ are the weights and $\odot$ represents an element-wise product.

The similarity network can be viewed as a binary classifier, and therefore we could use binary cross-entropy (BCE) as the loss function:

$$\begin{aligned} \mathcal{L}_{sim} = &- \Sigma_i^N y_i \cdot \log p(y_i) \\ &+ (1 - y_i) \cdot \log(1 - p(y_i)) \end{aligned} \tag{5}$$

The last layer of the similarity network is a fully-connected layer with a softmax activation function, and the output is an array of size two, in which the elements denote the weight for each class. We compute the matching score by multiplying $+1$ (matched) and $-1$ (unmatched) on these two elements:

$$\begin{aligned} \alpha &= \text{softmax}(\text{fc}(X_J)) \\ match(G,I)_{sim} &= 1 \cdot \alpha[0] + (-1) \cdot \alpha[1] \end{aligned} \tag{6}$$

where fc stands for fully-connected layer.

### A.1.3 Triplet Network

A Triplet Network requires the input to be in the format of a triplet $(G, I_{pos}, I_{neg})$. There will be three branches in the network which map the three embeddings to the same joint span:

$$\begin{aligned} \hat{X}_G &= L_2N(X_G W_{G\rightarrow J}) \\ \hat{X}_{I_{pos}} &= L_2N(X_{I_{pos}} W_{I\rightarrow J}) \\ \hat{X}_{I_{neg}} &= L_2N(X_{I_{neg}} W_{I\rightarrow J}) \end{aligned} \tag{7}$$

in which, $W_{G\rightarrow J} \in \mathbb{R}^{d_G \times d_J}$ and $W_{I\rightarrow J} \in \mathbb{R}^{d_I \times d_J}$ are weights, and the branches of positive and negative images share the same weight.

The network learns the cross-modality by minimizing the distance between positive pairs and maximizing the distance between negative pairs. We choose cosine distance as the distance function which will also be used to compute the matching score:

$$\begin{aligned} \mathcal{L}_{trip} = &\max(0, cos(\hat{X}_G, \hat{X}_{I_{pos}}) \\ &- cos(\hat{X}_G, \hat{X}_{I_{neg}}) + \text{m}) \\ match(G, I)_{trip} &= cos(\hat{X}_G, \hat{X}_I) \end{aligned} \tag{8}$$

Where m is the margin, which is set to $0.2$ in the experiment.

### A.1.4 LXMERT

LXMERT (Tan and Bansal, 2019) is a multimodal encoder that aims to ground text to images. It takes as an input image $I$ and a related sentence $G = \{w_1, w_2, \ldots, w_n\}$. The image objects are embedded using a feature extractor (Anderson et al., 2018) pre-trained on ImageNet (Deng et al., 2009). Given $I$ the detector finds $m$ objects $\{o_1, o_2, \ldots, o_m\}$ where: $o_i = \{p_i, f_i\}$, s.t. $p_i$ is its bounding box and $f_i$ is its 2048-dimensional region of interest (RoI).

LXMERT learns a position-aware embedding as follows:

$$f'_i = L_2N(W_F f_i + b_F) \quad (9)$$
$$p'_i = L_2N(W_P p_i + b_P) \quad (10)$$
$$v_i = (f'_i + p'_i)/2 \quad (11)$$

The text tokens are extracted using a tokenizer (Wu et al., 2016) and converted to index-aware embeddings s.t. $w_i$ and $i$ are projected onto embedding spaces $w'_i$, $u'_i$, to get a common embedding.

$$h_i = L_2N(w'_i + u'_i) \quad (12)$$

Those inputs are then passed through a language encoder $E_G$, an object relationship encoder $E_I$, and a cross-modality transformer encoder $E_J$. Let $X_I = \{v_1, v_2, \ldots, v_n\}$ and $X_G = \{h_1, h_2, \ldots, h_n\}$.

$$\begin{aligned} \hat{X_G} &= E_G(X_G) \\ \hat{X_I} &= E_I(X_I) \\ X_{G_J}, X_{G_I} &= E_J(\hat{X_G}, \hat{X_I}) \end{aligned} \quad (13)$$

Then the cross-modality output $X_J$ is extracted from the output embedding $X_{G_J}$ that corresponds to the special token [CLS] appended to each input text.

Similarly to A.1.2, we use BCE loss.

$$\begin{aligned} \mathcal{L}_{lxmert} = &- \Sigma_i^N y_i \cdot \log p(y_i) \\ &+ (1 - y_i) \cdot \log(1 - p(y_i)) \end{aligned} \quad (14)$$

and compute the matching score:

$$\begin{aligned} \alpha =& \text{softmax}(\text{fc}_2\text{fc}_1(X_J)) \\ match(G, I)_{lxmert} =& \ 1 \cdot \alpha[0] + (-1) \cdot \alpha[1] \end{aligned} \quad (15)$$

### A.2 Features

#### A.2.1 Vision

We select InceptionV3 (Szegedy et al., 2015) as the feature extractor for the image. We have tried VGG19 and Resnet50, but InceptionV3 turns out to have the best performance. We use the second last hidden layer of InceptionV3 to obtain a vector of (2048, ).

#### A.2.2 Language

We use a pre-trained BERT sentence transformer (Reimers and Gurevych, 2019) with `bert-base-uncased` as our base model. Then, we use max-pooling to get the feature vector with a dimension of (768, ).

| Model | Optimizer | Learning Rate | Batch Size | n. of Parameters |
|---|---|---|---|---|
| DeViSE | RMSProp | 5e-6 | 1024 | 2,897,664 |
| Similarity Net | RMSProp | 5e-6 | 1024 | 4,424,170 |
| Triplet Net | Adam | 1e-5 | 1024 | 4,984,832 |
| LXMERT | Adam | 5e-7 | 32 | 209,124,098 |

Table 5: Hyper Parameters of All Models.

### A.3 Hyper Parameters

See Table 5.

### A.4 Training Details

The training of DeViSE, Similarity Network and Triplet Network were on a single NVIDIA RTX 2080 for 200 epochs with early stopping. The training took less than 10 hours.

We used a pre-trained LXMERT model with 9 language layers, 5 cross-encoder layers, 5 vision encoder layers, and a 2 layer linear classification head, with `GELU()` (Hendrycks and Gimpel, 2016) and `ReLU()` activation, with a Sigmoid final layer and with normalization in the first layer.

We fine-tune the model for 10 epochs while allowing the gradient to flow through the LXMERT pre-trained layers. We use a binary cross-entropy loss from the PyTorch library and an Adam (Kingma and Ba, 2014) optimizer. Note that we deal with imbalanced datasets by repeating the positive samples and shuffling the data.

## B Goal-Image Retrieval Task

### B.1 Sampling

Goal-Image Retrieval is a more practical format that gives a high-level goal and a pool of images and aims to rank these images based on their similarity with the goal query.

In this experiment, we randomly select 1,000 high-level goals from the testing set of multiple-choice tasks and choose 5 images for each goal, thus building a pool of 5,000 images.

### B.2 Evaluation Metrics

We perform recall at $k$ (recall@$k$, higher the better) and median rank (Med r, lower the better) to measure the retrial performance. For the 5k image pool, $k \in \{10, 25, 50, 100\}$, while for the 1k image pool, $k \in \{1, 5, 10, 25\}$.

### B.3 In-Domain Performance

As shown in Table 6, the triplet network with BERT as the text embedding has the best performance.

| Model | 5K Testing Images | | | | |
|---|---|---|---|---|---|
| | R@10 | R@25 | R@50 | R@100 | Med r |
| Random | 0.1 | 0.4 | 1.0 | 2.1 | 2519 |
| DeViSE | 5.2 | 9.8 | 15.2 | 23.8 | 429 |
| Similarity Net | 5.8 | 11.5 | 17.6 | 27.0 | 347 |
| Triplet Net (GloVe) | 5.9 | 12.2 | 19.9 | 31.2 | 264 |
| Triplet Net (BERT) | **6.9** | **13.8** | **21.9** | **32.7** | **249** |

Table 6: In-Domain Retrieval results with Different Models.

| Prompt | Token Length | Vocab Size | 1K Testing Images | | | | |
|---|---|---|---|---|---|---|---|
| | | | R@1 | R@5 | R@10 | R@25 | Med r |
| Goal | 3.34 | 19,299 | 4.6 | 14.6 | 22.8 | 36.3 | 49 |
| Method | 3.11 | 24,180 | 2.5 | 11.4 | 18.9 | 33.3 | 57 |
| Step | **4.67** | **49,999** | **6.1** | **20.1** | **31.4** | **48.7** | **26** |

Table 7: Query on Different Prompts

## B.4 Query on Different Prompts

As can be seen from Table 7, the model has higher performance when using the detailed step description as a prompt. Through qualitative analysis (see Figure 8) on some samples, we discovered that some method descriptions are very general, and short abstract keywords are even more refined than the goal description. To quantify this finding, we calculate the average length of tokens (remove stop words) and the vocabulary size of the three types of prompts. Apparently, the step description is more fruitful than the method and goal with higher token length and vocab size. The method described has a lower average length of tokens, which is in line with our observation.

## B.5 Transfer Performance on Retrieval

We also evaluate the transfer performance on a retrieval task. For COIN, we choose 5-6 images for each video from the 180 goals and construct a pool of 1,000 images. For Howto100m, we randomly select 5-6 images of each of the videos in the testing set and also form a pool of 1K images.

Table 8 and 9 indicates the model pre-trained on wikiHow outperforms the other dataset in the retrieval task and the aggregation model could further improve the performance.

# C Step-Aggregation Model

We have seen that SOTA models do not perform well in VGSI because of the implicit vision-language relation. So we develop a step aggregation model that takes advantage of the existing goal-step knowledge from wikiHow. The main idea is as follows: given an unseen textual goal, we use $k$-nearest neighbors to find the most related

| PT-Data | 1k Test Images | | | | |
|---|---|---|---|---|---|
| | R@1 | R@5 | R@10 | R@25 | Med r |
| - | 0.0 | 0.5 | 1.2 | 2.5 | 517 |
| Flickr | 1.2 | 4.0 | 7.1 | 14.2 | 240 |
| MSCOCO | 0.9 | 5.5 | 9.3 | 19.0 | 170 |
| wikiHow | **1.4** | **7.6** | **12.6** | **23.8** | **102** |

Table 8: Zero-shot Retrieval on COIN

| PT-Data | FT? | 1K Test Images | | | | |
|---|---|---|---|---|---|---|
| | | R@1 | R@5 | R@10 | R@25 | Med r |
| - | ✓ | 1.1 | 4.4 | 9.5 | 17.4 | 129 |
| Flickr30K | ✗ | 0.9 | 3.9 | 6.9 | 11.7 | 213 |
| | ✓ | 1.2 | 5.4 | 10.5 | 20.9 | 122 |
| MSCOCO | ✗ | 0.5 | 4.1 | 7.3 | 13.8 | 202 |
| | ✓ | 1.7 | 6.8 | 11.9 | 22.4 | 98 |
| COIN | ✗ | 1.1 | 4.2 | 7.7 | 15.3 | 193 |
| | ✓ | 1.6 | 6.1 | 11.7 | 21.6 | 118 |
| wikiHow | ✗ | **1.6** | **7.3** | **13.5** | **25.1** | **88** |
| | ✓ | **2.0** | **7.9** | **14.7** | **26.7** | **84** |

Table 9: Retrieval Performance on Howto100m

article title from wikiHow, then extract the $n$ steps from this article as $S = \{s_1, s_2, ..., s_n\}$. Instead of directly using the given goal to match the images (goal score - $\text{Score}_g$), we could use the sequence of steps to improve the matching (step score - $\text{Score}_s$). Then use linear interpolation to summarize these two scores as our final matching score.

$$\begin{aligned} \text{Score}_g &= match(G, I) \\ \text{Score}_s &= \max_{i=1:n}(match(s_i, I)) \\ \text{Score}_{final} &= \lambda \cdot \text{Score}_g + (1-\lambda) \cdot \text{Score}_s \end{aligned} \tag{16}$$

where, $\lambda$ adjusts the step and goal scores weights, we choose $\lambda = 0.5$.

The main idea of the model is to break down the high-level goal into intermediate steps via schema. Then we use the induced sequence of steps as the new query to improve the matching performance. For example in Figure 6, when we want to match the goal "Install License Plate" with two images, the model makes a wrong choice because the negative sample (the right one) also involves the "install" action. However, we could fetch the intermediate steps from wikiHow and use these steps to match the images. The left image (the correct choice) has a higher Step-Image similarity score than the right one. Therefore, the model could improve its performance with the help of this step information. As we can see from the example steps, they contain some useful entities such as "screw", "bracket", "bumper", etc., which are closely related

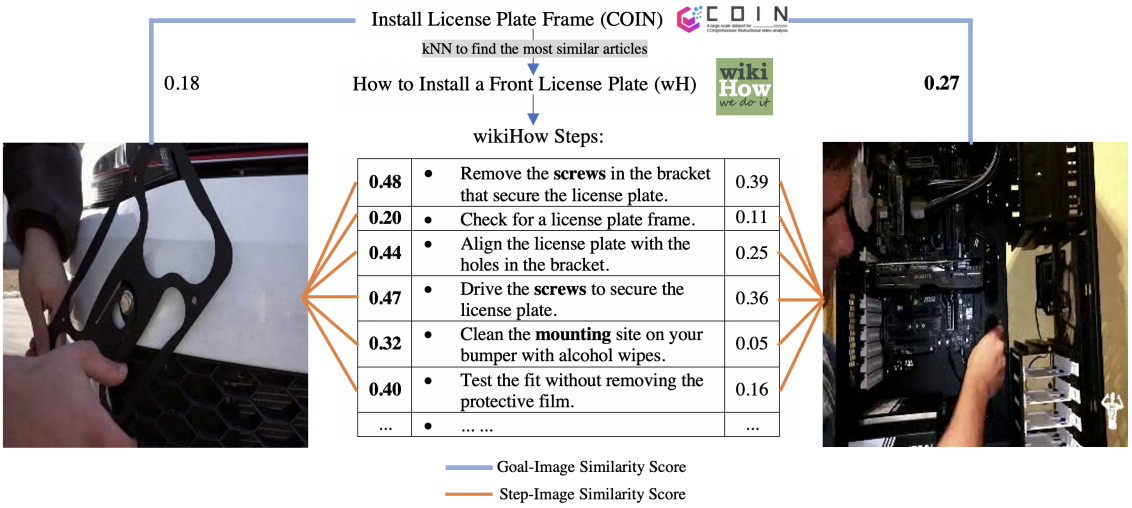



Figure 6: The architecture of the Step-Aggregation Model.

| Dataset | Model | Sampling Strategy | | |
|---|---|---|---|---|
| | | Random | Similarity | Category |
| COIN | wikiHow | .7639 | .6854 | .5659 |
| | wikiHow$^{agg}$ | **.7657**(+0.2%) | **.6942**(+1.3%) | **.5764(+1.9%)** |
| Howto100m | wikiHow | .6855 | .7249 | .5143 |
| | wikiHow$^{agg}$ | **.6947**(+1.3%) | **.7392(+2.0%)** | **.5245(+2.0%)** |

Table 10: Apply Step-Aggregation model on multiple-choice VGSI ($agg$ stands for aggregation model).

| Dataset | Model | R@1 | R@5 | R@10 | R@25 | Med r |
|---|---|---|---|---|---|---|
| COIN | wikiHow | 1.4 | 7.6 | 12.6 | 23.8 | 102 |
| | wikiHow$^{agg}$ | **1.9**(+35.7%) | **7.8**(+2.6%) | **13.6**(+7.9%) | **25.9**(+8.8%) | **97**(-4.9%) |
| Howto100m | wikiHow | 2.0 | 7.9 | 14.7 | 26.7 | 84 |
| | wikiHow$^{agg}$ | **2.1**(+5.0%) | **8.3**(+5.1%) | **15.8**(+7.5%) | **27.7**(+3.7%) | **80**(-4.8%) |

Table 11: Apply Step-Aggregation model on retrieval VGSI.

to the visual information in the image but do not show up in the goal sentence.

We apply the aggregation model on both multiple-choice and retrieval VGSI tasks. As shown in Table 10 and 11, with the assistance of the aggregation model, the accuracy of multiple-choice increased by 0.2% - 2%, and the median rank of retrieval decreased by 5%. Since our approach to utilize these steps is very simple, but still achieve a marginal improvement. We hope to see more advanced models to realize the full potential of wikiHow steps.

## D Qualitative Examples

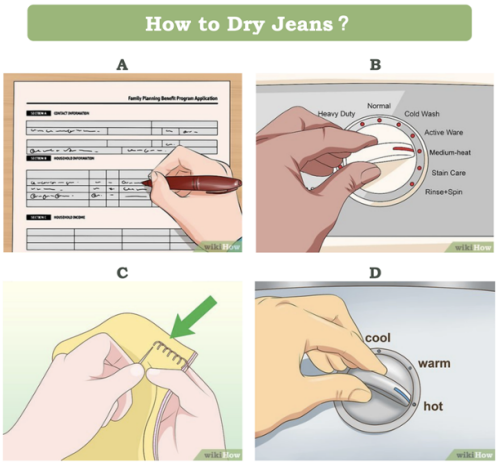
See Figure 7, 8, 9.

(A) Random Sampling


(A.1) Correct Answer is **B**


(A.2) Correct Answer is **A**

(B) Similarity Sampling




(B.1) Correct Answer is **B**


(B.2) Correct Answer is **C**

(C) Category Sampling


(C.1) Correct Answer is **A**


(C.2) Correct Answer is **D**

Figure 7: Qualitative Examples Using Different Sampling Strategies.

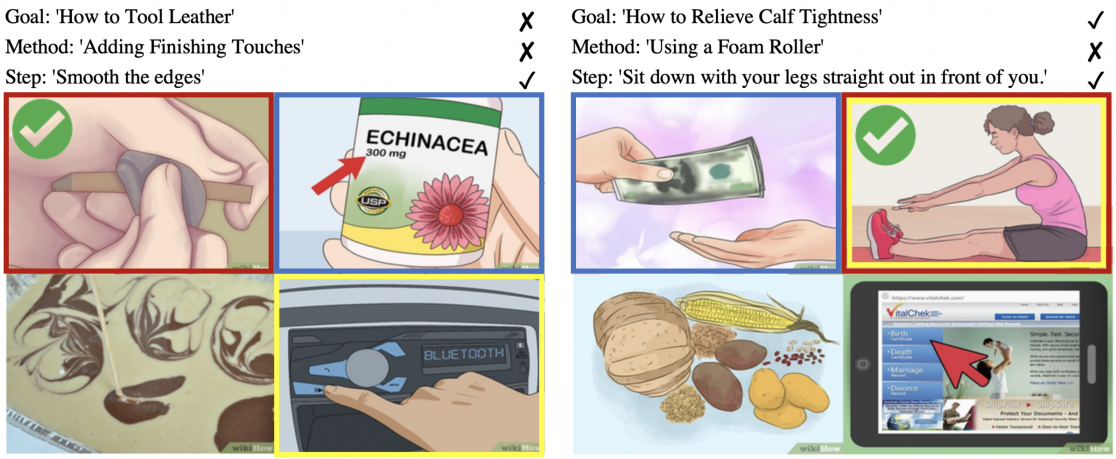



Figure 8: Qualitative Examples Using Different Query Prompts. (Yellow bounding box is the goal's prediction, blue bounding box denotes the method's prediction, red bounding box denotes the step's prediction, green checkmark represents the ground truth.)

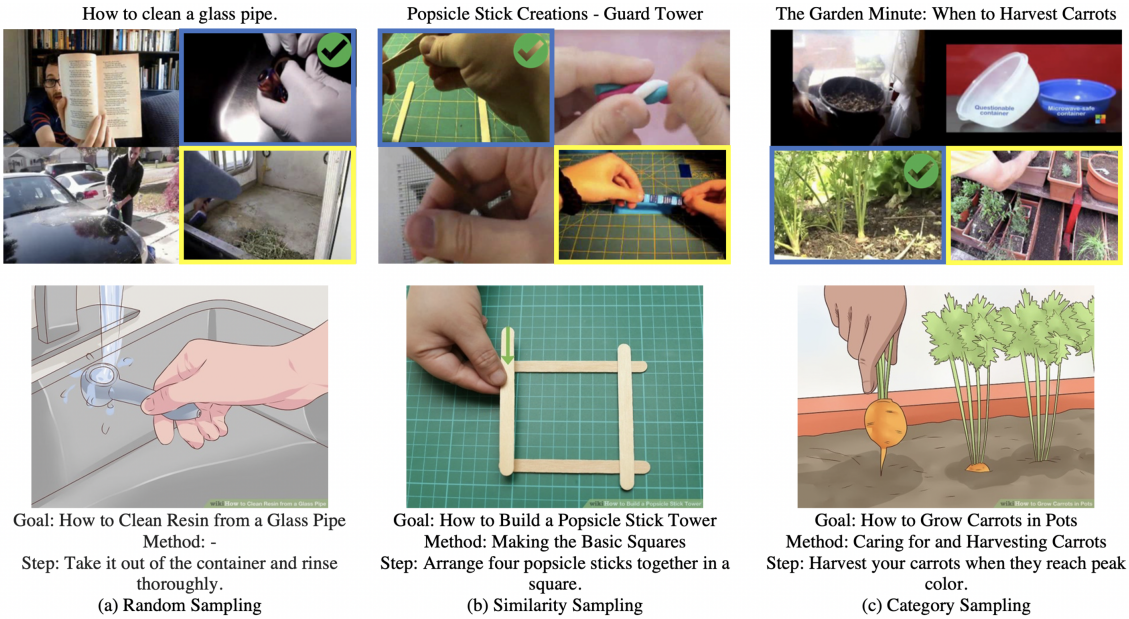



Figure 9: Qualitative Examples of Transfer Learning on Howto100m. (The first row shows the multiple-choice examples of Howto100m video frames, the yellow bounding box is the prediction of the model without pre-training on wikiHow, blue bounding box denotes the prediction of the pre-trained model, and green checkmark represents the ground truth. The second row shows the related images and descriptions we found in wikiHow.)