A. Tokenizer

Given the raw inputs from text, image, and video modalities, modality-specific tokenizers are applied to generate the input token sequences for the Transformer encoder. Here, we use the BPE tokenizer [67] for text modality, the image patch tokenizer [20] for image modality, and the temporal frame patch tokenizer [7] for video modality. These outputted tokens are attached with additional modality type embeddings to identify which modality the raw input belongs to. The tokenizers are illustrated in Fig. 2.

Text Tokenizer. The BPE Tokenizer [67] is employed for text modality. The text inputs are split into sub-words and projected by a linear embedding layer. A learnable 1D positional embedding for text with a max length of 256 is added to the word embeddings. To specify the input modality, an additional trainable textual modality embedding <T> is added to each token embedding. Note that all the text inputs of our model share the same vocabulary.

Image Tokenizer. The image patch tokenizer [20] is utilized as the image tokenizer. The input images are resized to 224×224, and flattened to a sequence of image patches with shapes 16×16, which are further mapped by a linear projection. A sequence of learnable image positional embeddings with a fixed length $14 \times 14 = 196$, and an additional visual modality embedding <V> are also added.

Video Tokenizer. The temporal frame patch tokenizer [7] is used for video tokenization. Each frame of the input video is flattened to image patches with shape 16×16. For a video with N frames ($N = 8$ by default), the number of tokens would be $N \times 14 \times 14$. The spatial positional embeddings for images as well as a 1D temporal position embedding with max length $N = 8$ are added to the video embeddings. Besides, the additional visual modality embedding <V> is added to each token embedding.

B. Implementation Details for Auto-regressive Language Modeling

In the formulation of autoregressive tasks from previous works, each input token attends to all previous tokens, including itself, to predict **the next word**. Unlike previous works, we use <SPE> to predict **the current word**. Fig. 4 shows how we achieve this. In the inference stage, <SPE> is appended to the end of the predicted tokens as the input. The corresponding output is used for prediction. In the training stage, we append several <SPE> tokens after the input sequence. Each <SPE> token is trained to predict a word in the input sequence. The attention mask is designed to make sure that the word tokens do not attend to <SPE> tokens, and each <SPE> token only attends to itself and the previous word tokens. In this way, the training stage and the inference stage are aligned.

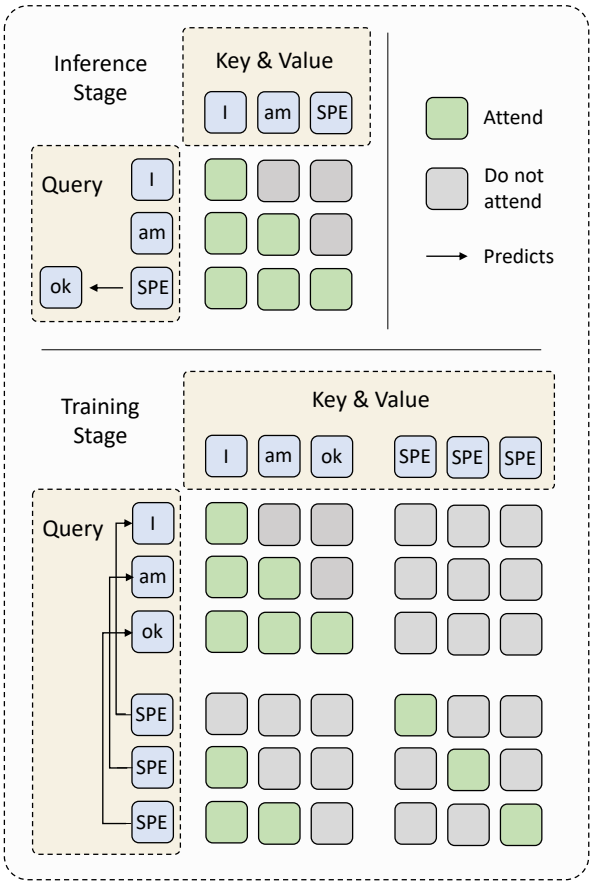

Figure 4. The attention mask of autoregressive language modeling.

C. Prompt Tuning

Four groups of parameters are learnable in prompt-tuning. They are <SPE>, layer normalization parameters, prompt tokens, and linear heads. This section introduces the usage and the number of parameters of each parameter group.

Details Similar to the pre-training stage, <SPE> token is shared for both inputs and targets. Layer normalization weights and biases in each Transformer layer and tokenizer are tuned. Learnable prompts are added to each layer of the Transformer encoder. Specifically, the input prompts of each layer do not come from the output of the previous layer. They are random initialized learnable parameters. For all prompt-tuning experiments, we use 10 learnable prompts for each layer on both inputs. The linear heads only apply for classification tasks. It takes the feature that is to be classified as input, and returns a classification logit. The output probability is a linear combination of the probability from the similarity score and the linear head:

$$P(x, y) \propto \alpha \exp\left(\cos\left(f(x), f(y)\right)/\tau\right) + w^{\top} f(x) + b \quad (4)$$

	Number of Parameters
<SPE>	768
Layer Norm	41,472
Prompt	184,320
Linear head	768 * num_classes

Table 9. The number of learnable parameters in the prompt-tuning stage. Note that the linear head only applies for classification tasks.

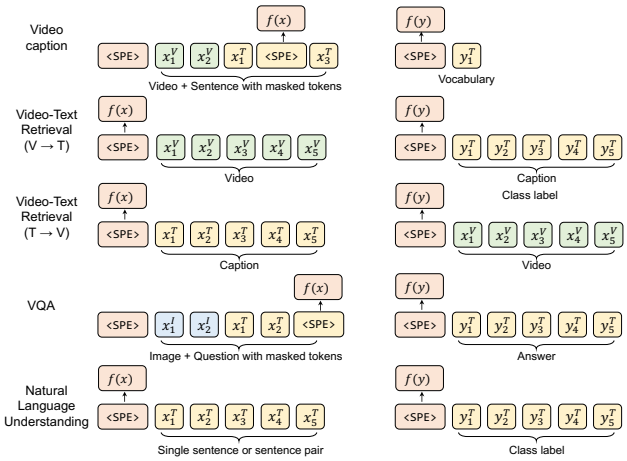

Figure 5. Input and target formats of our novel tasks. For each task, the left column represents the format of input sequence x, and the right column represents the format of the target sequence y. $f(x)$ and $f(y)$ are used to calculate the joint probability distribution. Here, we have omitted the tokenizer and encoder for concision.

where w and b are the weights and bias of the linear, and α is a learnable scalar. w and b are initialized with 0 and α is initialized with 1.

Number of Parameters Tab. 9 shows the number of parameters of each learnable component in prompt-tuning. For tasks other than classification, the number of parameters is 227K in total. When the linear head is added, *e.g.*, in ImageNet-1k classification task, the number is 995K, which is still less than 1% of all the parameters in the pre-training stage.

D. Formulation of Novel Tasks

The generic perceptual modeling makes it easy to convert existing tasks into the unified task formulation of our Uni-Perceiver. Fig. 5 illustrates the input and the output formulations of our novel tasks, which we will describe in details.

Video Caption. Similar to image caption, video caption is modeled as the autoregressive language modeling task with video clues. In this task, the model predicts each word based on the video and its previous words. The input set $\mathcal{X}$ consists of the sequence concatenation of video and the words that have been predicted, followed with a <SPE> token. The output features of the <SPE> token is used to calculate the joint probability with each words in the the vocabulary set $\mathcal{Y}$. Then the word with the highest probability is the predicted word at the current location. Additionally, video caption follows the efficient implementation of autoregressive training introduced in Sec. B.

Video-Text Retrieval. The Video-Text retrieval follows the formulation of Image-Text retrieval task, except that the image sequence is replaced by the video sequence. Specifically, the input sets $\mathcal{X}$ and $\mathcal{Y}$ are composed of video and text sequences respectively. Each sequence in $\mathcal{X}$ and $\mathcal{Y}$ also has a <SPE> token at the beginning. We use the output feature at the <SPE> token as the final representation of the input video or the text to calculate the joint probability distribution.

Visual Question Answering. We formulate the VQA task as a special case of masked language modeling with image clues. The input $x \in \mathcal{X}$ is the combination of image and question sequences. It should be noted that each question ends with a "?" mark and a <SPE> token is appended to the end of the question sequence. The $\mathcal{Y}$ set consists of candidate answer sequences, in which each begins with a <SPE> token too. The joint probability distribution between $\mathcal{X}$ and $\mathcal{Y}$ can be calculated by using the output feature from <SPE> tokens.

Natural Language Understanding. The formulation of natural language understanding task is similar to that of image classification task. $\mathcal{X}$ denotes the set of the input single sentence or the sentence-pair, and $\mathcal{Y}$ is the set contains the textual class label. For example, in SST-2 [71], x denotes the sentence sequence of movie review and $y \in \mathcal{Y} = \{\texttt{great}, \texttt{terrible}\}$ is the sentiment label. In MRPC [19], x instead is the sequence combination of sentence pairs extracted from news, and $y \in \mathcal{Y} = \{\texttt{Yes}, \texttt{No}\}$ is the label to indicate whether the sentence pair are semantically equivalent. We also add <SPE> tokens at the beginning of the sequences x and y, of which output features are used to computed joint probability.

E. Extra pre-training details

Sampling Weight & Batch Size & Loss Tab. 10 lists the batch size and sampling weight of each task and dataset in the pre-training stage. We use cross-entropy loss for language modeling tasks. The other tasks are trained with cross-entropy loss with 0.1 label smoothing. The loss weight of video classification is 0.05, which helps stabilize training in our experiments. The other loss weights are 1.0 by default.

For retrieval tasks like image-text retrieval, we use train-

Task	Dataset	Batch Size	Sampling Weight
Image Classification	ImageNet-21k [17]	64	0.333
Video Classification	Kinetics-700 [32]	4	0.0925
	Moments in Time [57]	24	0.0185
Auto-encoding LM	Books&Wiki [93]	64	0.07775
	YFCC [31]	64	0.02778
	CC12M [9]	64	0.02778
	CC3M [68]	64	0.01389
	Visual Genome [36]	64	0.01389
	COCO Caption [12]	64	0.01389
	SBU [58]	64	0.01389
	PAQ [41]	512	0.0222
Auto-regressive LM	Books&Wiki [93]	64	0.07775
	YFCC [31]	56	0.02778
	CC12M [9]	56	0.02778
	CC3M [68]	56	0.01389
	Visual Genome [36]	56	0.01389
	COCO Caption [12]	56	0.01389
	SBU [58]	56	0.01389
	PAQ [41]	400	0.0222
Retrieval	YFCC [31]	128	0.02778
	CC12M [9]	128	0.02778
	CC3M [68]	128	0.01389
	Visual Genome [36]	128	0.01389
	COCO Caption [12]	128	0.01389
	SBU [58]	128	0.01389
	PAQ [41]	512	0.0222

Table 10. Ingredients and hyper-parameters for our pre-training.

ing samples in the same batch as negative samples, whose typical size is 127 except the PAQ dataset. Note that we do not use memory bank and do not gather feature across GPU devices to provide more negative samples, which may further promote the performance of retrieval tasks.

Data Augmentation We apply augmentation techniques to image and video modalities to avoid overfitting. For images in ImageNet-21k dataset, we apply augmentation same as [77]. Rand-Aug[16], random erasing [89], mixup [86] and cutmix [85] are used simultaneously. For images in other datasets, we resize the images to the short edge size of 256, and then a 224×224 region is cropped randomly from the resized images during training. During inference, the random crop is replaced with center crop operation. For all Video inputs, we apply the same augmentations used in [7]. We use clips of size $8 \times 224 \times 224$ for Kinetics-700 and Kinetics-400, and $3 \times 224 \times 224$ for Moments in Time. The temporal sample rate is 32. We use a single temporal clip. During training, the start frame is randomly picked if the video is longer than the clip. In the training stage, we first resize the shorter side of the video to a random value in $[256, 320]$, then we randomly sample a 224×224 crop. In the test stage, the short side of the video is resized to 224. For classification tasks, We use 3 spatial crops with size 224×224 to cover a larger range of content and average their logits for evaluation. For video captioning task, we use center crop after resizing.

Data Parallel for Vocabulary and Class Labels Naive implementation of language modeling and ImageNet-21k classification is impractical due to memory limitation. In language modeling, we need to compare the feature of a <SPE> token in a sequence to the feature of each token in our tokenizer. Since the vocabulary size is large, we apply data parallel for the vocabulary set. A similar method is used for class labels of ImageNet-21k.

Removing Overlap For the training set of K700 participating in pre-training, we remove those videos overlapping with validation set of K400.

F. Licences of Datasets

ImageNet-21K [17] is subject to the ImageNet terms of use [79].

Kinetics-700 [70] & **Kinetics-400** [32] The kinetics dataset is licensed by Google Inc. under a Creative Commons Attribution 4.0 International License.

BooksCorpus [93] Replicate Toronto BookCorpus is open-source and licensed under GNU GPL, Version 3.

Wikipedia Most of Wikipedia's text is co-licensed under the Creative Commons Attribution-ShareAlike 3.0 Unported License (CC BY-SA) and the GNU Free Documentation License (GFDL) (unversioned, with no invariant sections, front-cover texts, or back-cover texts). Some text has been imported only under CC BY-SA and CC BY-SA-compatible license and cannot be reused under GFDL.

YFCC [31] All the photos and videos provided in YFCC dataset are licensed under one of the Creative Commons copyright licenses.

CC12M [9] is licensed under the Terms of Use of Conceptual 12M [52].

CC3M [68] is licensed under the Conceptual Captions Terms of Use [53].

Visual Genome [36] is licensed under a Creative Commons Attribution 4.0 International License [35].

COCO Caption [12] The images are subject to the Flickr terms of use [22].

SBU Caption [58] The images are subject to the Flickr terms of use [22].

PAQ [41] is licensed under the Attribution-NonCommercial 4.0 International License.