Action Class Relation Detection and Classification Across Multiple Video Datasets

Yuya Yoshikawa^{a,*}, Yutaro Shigeto^a, Masashi Shimbo^a, Akikazu Takeuchi^a

^aSoftware Technology and Artificial Intelligence Research Laboratory., Chiba Institute of Technology, Tsudanuma 2-17-1, Narashino, 275-0016, Chiba, Japan

Abstract

The Meta Video Dataset (MetaVD) provides annotated relations between action classes in major datasets for human action recognition in videos. Although these annotated relations enable dataset augmentation, it is only applicable to those covered by MetaVD. For an external dataset to enjoy the same benefit, the relations between its action classes and those in MetaVD need to be determined. To address this issue, we consider two new machine learning tasks: action class relation detection and classification. We propose a unified model to predict relations between action classes, using language and visual information associated with classes. Experimental results show that (i) pre-trained recent neural network models for texts and videos contribute to high predictive performance, (ii) the relation prediction based on action label texts is more accurate than based on videos, and (iii) a blending approach that combines predictions by both modalities can further improve the predictive performance in some cases.

Keywords: Relation prediction, MetaVD, Human action recognition, Multi-modal classification

1. Introduction

Human action recognition (HAR) in videos has a wide variety of applications, such as content-based video summarization [1], video retrieval [2], and video surveillance [3]. It is an active research topic in computer vision (CV), and many practical HAR datasets have been made publicly available [4] by the research community. However, as these datasets were developed for specific purposes and needs of individual researchers, each dataset often consists of videos from a limited domain. When a model is trained with such a limited dataset with insufficient diversity, it often fails to correctly recognize videos from different domains than used for training.

To help improve the robustness of HAR models over di-

verse domains, Yoshikawa et al. [5] constructed the Meta Video Dataset (MetaVD), a meta-dataset of six existing HAR datasets: UCF101 [6], HMDB51 [7], ActivityNet (v.1.3) [8], STAIR Actions (v.1.1) [9], Charades [10], and Kinetics-700 [11]. As each dataset defines different classes of actions, MetaVD provides a curated list of related action classes across datasets. Specifically, three types of relations can be associated with pairs of actions: equal, similar, and is-a.

The relation labels in MetaVD naturally enable dataset augmentation among the datasets. For example, suppose that action class i in a dataset ("target dataset") is labeled as equal to action class j in the other datasets ("source datasets"). Then, to train an HAR model for the target dataset, we can use the videos associated with class j in the source datasets as augmented training data for class i. The resulting model is expected to be more accurate and robust, as it is trained with data not only larger in size but also more diverse. Indeed, Yoshikawa et al. [5] report that in terms of test accuracy on the expanded target dataset,

^{*}Corresponding author.

Email addresses: yoshikawa@stair.center (Yuya Yoshikawa), shigeto@stair.center (Yutaro Shigeto), shimbo@stair.center (Masashi Shimbo), takeuchi@stair.center (Akikazu Takeuchi)



Figure 1: Illustration of action class relation detection and classification.

the HAR models trained on the expanded target dataset have substantially higher recognition accuracy than those trained on the original target dataset alone. Moreover, augmentation did not degrade the accuracy on the original target data, thus making the learned model robust to more diverse data without adverse side effects.

Despite its effectiveness, dataset augmentation using MetaVD has an obvious limitation: It can only be applied to the six datasets in MetaVD. To augment an external dataset in the same manner (i.e., using the external dataset as the target dataset and the entire MetaVD as the source dataset), one first needs to determine the relationship between the action classes in the dataset and those in MetaVD. Automating this process is highly desirable, as manual annotation of relations over all possible pairs of actions is time-consuming and expensive.

In this paper, we consider two supervised machine learning tasks that are relevant to this automation: *action class relation detection* and *action class relation classification*. Fig. 1 illustrates these tasks. Action class relation detection consists of predicting whether a pair of action classes (or *actions* for short) are related or not, while action class relation classification consists of predicting the relation type of two actions that are known to be related. The observed data in these tasks are a collection of videos with action labels and a collection of relation labels defined in MetaVD. They also have the aspect of multi-modal learning, because the action classes are characterized by visual information in the videos and the language information in the action labels.

We propose a unified model for these tasks. The in-

put to the model is a pair of actions, represented either by their text labels or by sets of videos showing the actions. The input actions are then individually transformed into embedding vectors by a suitable encoder for the input modality (text or video). After concatenating the embedding vectors for the two actions, the model outputs a prediction of the relation for the pair through a task-specific head. Furthermore, we explore simultaneously using both modalities using a blending approach, which outputs the weighted sum of the predictions from the text label of actions and the video sets.

Through the empirical evaluation of the proposed model, we answer the following questions.

- Can we build practical models for relation prediction using existing pre-trained neural network models for CV and natural language processing (NLP)?
- Which modality of language and vision is better suited to relation prediction?
- Can we improve prediction performance by using both modalities simultaneously rather than using only one?

We evaluate whether the model accurately predicts the relations between the target dataset and the source datasets in the two tasks, using five of the six datasets in MetaVD as source datasets and the remaining one as the target dataset. The experimental results reveal that the action label texts are consistently more useful than the videos in the two tasks. We also found that the performance of the blending approach is better than using only one modality in some cases. However, when the performance in the video modality is low, the blending approach tends to perform worse than using only the text modality.

2. Related Work

Relation prediction appears in various research fields. A typical example is knowledge base completion (KBC), which is the task of finding unknown relations between entities in a knowledge base. For KBC, numerous studies take the approach that represents entities and relations as embedding vectors or matrices [12], which are learned from a lot of triplets (head, relation, tail) in training data, where head and tail denote entities in the knowledge base. To obtain good representations of the embedding vectors and matrices, several studies enriched them by exploiting auxiliary information of the entities, such as textual description [13] and images [14]. By representing each action as an entity, our tasks can be regarded as the KBC that embodies the entity with an action label text and a set of videos. Unlike the standard KBC, our tasks require predicting a relation between the entity appearing in training data and that appearing only in test data. Such a problem is called *out-of-knowledge base entity problem* in the KBC literature [15].

Another relation prediction task in CV and visionand-language is visual relation detection (VRD), which is the task of recognizing relations or interactions between objects in an image [16]. For example, in VRD, the relations are represented in the form of $\langle object1, predicate, object2 \rangle$, and the goal is to predict the predicate from the visual information, e.g., the visual representation of the objects, and the linguistic information, e.g., the class names of the objects. For example, a visual translation embedding network [17] models the relations such that the sum of the low-dimensional representations of object1 and the predicate is equal to that of object2. Also, VRD studies that specialize in human-object interaction in which the predicate always represents actions have been conducted [18]. Furthermore, beyond images, several studies have focused on video visual relation detection (VVRD) [19]. Here, the visual features of an object in VVRD are obtained by spatio-temporally tracking the object in a video. However, our relation prediction tasks differ from VRD and VVRD, as our study aims to predict the relationships between actions rather than objects.

3. Action Class Relation Detection and Classification

3.1. Meta Video Dataset (MetaVD)

As introduced in Section 1, MetaVD annotates related actions across six major HAR datasets [5]. The following three types of relations are defined from the linguistic perspective on the text labels of action classes:

- equal Action labels A and B have the same meaning; e.g., "drink" and "drinking," and "smile" and "laugh."
- similar Action labels A and B have similar meanings; e.g., "stroking animal" and "grooming dog/horse."

is-a Action label A is a subordinate concept of action label B; e.g., "smoking hookah" and "smoking." Note that label A (hyponym) in this example corresponds to to_action_name in MetaVD, and label B (hypernym) corresponds to from_action_name.

The number of unique action classes in MetaVD, i.e., the union of sets of action classes in the six HAR datasets, is 1,309. There are 56,015 unique pairs of action classes across different datasets. Of these pairs, 320, 1,470, and 1,010 are labeled as equal, similar, and is-a, respectively. The remaining 565,214 are implicitly deemed unrelated.

As described in Section 1, MetaVD can be used for dataset augmentation to improve the accuracy of HAR in wider domains. However, currently, only the datasets within MetaVD can take advantage of this benefit. To allow datasets outside of MetaVD to benefit as well, we address two new tasks presented in the next subsection.

3.2. Two New Tasks Towards Automatic Relation Annotation

To augment an external dataset with MetaVD, the relationship must be identified between actions in the dataset and MetaVD. To address this need, we consider two machine learning tasks, which we refer to as action class relation detection and action class relation classification. Fig. 1 illustrates these tasks. Action class relation detection is the task of predicting whether or not a pair of actions is related, i.e., has any of equal, similar, and is-a in the original MetaVD relation types. Action class relation classification aims to predict which relation type holds for a pair of actions that are known to be related. It is a task of multi-class classification into four relation types: equal, similar, subclass-of, and superclass-of. Here, the original is-a relation defined in [5] is split into subclass-of and superclass-of, because we want to tell which of the action pair is a subordinate to the other.

We divide the six datasets in MetaVD into the source set that consists of five datasets, and the target set that consists of the remaining one dataset. The intention is to simulate a scenario in which the target set is the external dataset that we want to augment, and the source set is the internal datasets in (reduced) MetaVD. Thus, the goal is to predict the relation between the actions in the source set and those in the target set.

We now give a formal definition of the tasks in this simulated setting. Each action *i* has a text label $\mathbf{x}_i^{(\text{label})}$ and a collection $\mathbf{x}_i^{(\text{video})}$ of *K* videos. Let C_{src} and C_{tar} be the disjoint sets of actions that occur in the source and target datasets, respectively. Denote the set of text labels for the source set by $\mathcal{D}_{\text{src}}^{(\text{label})} = {\mathbf{x}_i^{(\text{label})} \mid i \in C_{\text{src}}}$ and its set of video collections by $\mathcal{D}_{\text{src}}^{(\text{video})} = {\mathbf{x}_i^{(\text{video})} \mid i \in C_{\text{src}}}$. The set $\mathcal{D}_{\text{tar}}^{(\text{label})}$ and $\mathcal{D}_{\text{tar}}^{(\text{video})}$ are defined likewise for the target set.

In both relation detection and classification, the models receive a pair of action classes, but their output is different.

In action class relation detection, the model must predict the presence or absence of a relationship between the input pair of actions. To be precise, when a pair of actions i, j has any of the relations equal, similar, and is-a in the sense of the original MetaVD relation types, the desired output is $y_{ij}^{(det)} = 1$ (positive), and $y_{ij}^{(det)} = 0$ (negative) otherwise. Thus, the positive action pairs in the source set is $\mathcal{R}^{(det)} = \{(i, j) \mid i, j \in C_{src}, i \neq j, y_{ij}^{(det)} = 1\}$. To train a model, $\mathcal{R}^{(det)}$ is divided into the training set $\mathcal{R}^{(det)}_{train}$ and the validation set $\mathcal{R}^{(det)}_{val}$ such that $\mathcal{R}^{(det)} = \mathcal{R}^{(det)}_{train} \cup \mathcal{R}^{(det)}_{val}$. After training, the model is used to predict whether any relationship exists between an action in the source set and one in the target set, i.e., each of the action pairs in the test set $\mathcal{R}^{(det)}_{test} = \{(i, j) \mid i \in C_{src}, j \in C_{tar}\}$.

In the action class relation classification task, the model must output one of the relation types among equal, similar, subclass-of, and superclass-of. Specifically, to train a model, we are given examples in the source set, i.e., $\mathcal{R}^{(\text{cls})} = \{(i, j, y_{ij}^{(\text{cls})}) \mid i, j \in C_{\text{src}}, i \neq j, y_{ij}^{(\text{det})} = 1\}$, where $y_{ij}^{(\text{cls})} \in \{0, 1\}^4$ is the one-hot ground-truth relation class vector. For example, when there is a similar relation between actions i, j, we set $y_{ij}^{(\text{cls})} = [0, 1, 0, 0]^{\top}$. Similarly to the relation detection task, the set $\mathcal{R}^{(\text{cls})}$ is divided into the training set $\mathcal{R}_{\text{train}}^{(\text{cls})}$ and the validation set $\mathcal{R}_{\text{val}}^{(\text{cls})}$, and used for training a prediction model. The trained model is applied to the test set given by $\mathcal{R}_{\text{test}}^{(\text{cls})} = \{(i, j) \mid i \in C_{\text{src}}, j \in C_{\text{tar}}, y_{ij}^{(\text{det})} = 1\}$.

4. A Unified Relation Prediction Model

4.1. Model Overview

We introduce a unified model for the two tasks described in Section 3, namely, action class relation detection and classification. An overview of the model is shown in Fig. 2. The model receives as input either the action label text $\mathbf{x}^{(label)}$ or the video set $\mathbf{x}^{(video)}$ for two actions $i, j \in C_{\text{src}} \cup C_{\text{tar}}$. We denote the input in modality $m \in \{\text{label}, \text{video}\}$ for action i by $\mathbf{x}_i^{(m)}$. The inputs $\mathbf{x}_i^{(m)}$ and $\mathbf{x}_j^{(m)}$ are fed into the encoder module $f_{\Theta}^{(m)}$ of modality m to obtain embedding vectors $\mathbf{h}_i^{(m)}, \mathbf{h}_i^{(m)} \in \mathbb{R}^{d_{\text{emb}}}$ as follows:

$$\boldsymbol{h}_{i}^{(m)} = f_{\Theta}^{(m)}\left(\boldsymbol{x}_{i}^{(m)}\right), \quad \boldsymbol{h}_{j}^{(m)} = f_{\Theta}^{(m)}\left(\boldsymbol{x}_{j}^{(m)}\right), \quad (1)$$

where Θ is a set of parameters for the encoder modules of both modalities. Then, to eliminate a bias in the input order of the actions, the embedding vectors are concatenated in two ways by swapping their order as follows:

$$\boldsymbol{c}_{ij}^{(m)} = \operatorname{concat}\left(\boldsymbol{h}_{i}^{(m)}, \boldsymbol{h}_{j}^{(m)}\right), \quad \boldsymbol{c}_{ji}^{(m)} = \operatorname{concat}\left(\boldsymbol{h}_{j}^{(m)}, \boldsymbol{h}_{i}^{(m)}\right),$$
(2)

where concat(\cdot, \cdot) concatenates two input vectors. Next, $c_{ij}^{(m)}$ and $c_{ji}^{(m)}$ are fed into the detection head $g_{\Phi}^{(det)}$ or the classification head $g_{\Phi}^{(cls)}$, depending on the task to be solved, where Φ is a set of parameters included in the heads. We formalize the detection head as a probabilistic binary classifier, that is,

$$g_{\Phi}^{(\text{det})}\left(\boldsymbol{c}_{ij}^{(m)}, \boldsymbol{c}_{ji}^{(m)}\right) = \sigma^{(\text{det})}\left(\mu^{(m,\text{det})}\left(\boldsymbol{c}_{ij}^{(m)}\right) + \mu^{(m,\text{det})}\left(\boldsymbol{c}_{ji}^{(m)}\right)\right),\tag{3}$$

where $\mu^{(m,det)}(\cdot)$ is a function that maps the input into a scalar value, which is, for example, defined as a multilayer perceptron (MLP) and a linear function, and $\sigma^{det}(\cdot)$ is the sigmoid function.

The classification head is formalized as a four-class probabilistic classifier, that is,

$$g_{\Phi}^{(\mathrm{cls})}\left(\boldsymbol{c}_{ij}^{(m)}, \boldsymbol{c}_{ji}^{(m)}\right) = \sigma^{(\mathrm{cls})}\left(\mu^{(m,\mathrm{cls})}\left(\boldsymbol{c}_{ij}^{(m)}\right) + \pi\left(\mu^{(m,\mathrm{cls})}\left(\boldsymbol{c}_{ji}^{(m)}\right)\right)\right),\tag{4}$$

where $\mu^{(m,cls)}(\cdot)$ is a function that maps the input into a four-dimensional vector, $\sigma^{(cls)}(\cdot)$ is the softmax function, and $\pi(\cdot)$ is a permutation function that exchanges only the dimensions corresponding to subclass-of and superclass-of. This serves to eliminate the effect of the input order of the actions. Finally, through the head,



Figure 2: Overview of our unified relation prediction model.



Figure 3: Architectures of the label encoder (top) and the video set encoder (bottom).

the model outputs a prediction of the relation between actions *i*, *j* for modality *m* in task $t \in \{\text{det, cls}\}$, denoted by $\tilde{y}_{ij}^{(m,t)}$. In particular, the prediction of the detection task is $\tilde{y}_{ij}^{(m,\text{det})} \in [0, 1]$, while that of the classification task is $\tilde{y}_{ij}^{(m,\text{cls})} \in [0, 1]^4$ such that the sum of $\tilde{y}_{ij}^{(m,\text{cls})}$ is equal to one.

4.2. Label and Video Set Encoders

The critical parts of the model are the label encoder and the video set encoder shown in Fig. 3. The label encoder $f_{\Theta}^{(\text{label})}(\cdot)$ is a deep neural network that outputs the embedding vector $\boldsymbol{h}^{(\text{label})} \in \mathbb{R}^{d_{\text{emb}}}$ from the action label text $\boldsymbol{x}^{(\text{label})}$. It first obtains a text representation vector by applying a text backbone network to the input action label text. It then transforms the hidden vector using an MLP and outputs the embedding vector $\boldsymbol{h}^{(\text{label})}$. The video set encoder $f_{\Theta}^{(\text{video})}(\cdot)$ is a deep neural network that outputs the embedding vector $\boldsymbol{h}^{(\text{video})}$ from a set of K videos $\boldsymbol{x}^{(\text{video})}$. First, the K videos in $\boldsymbol{x}^{(\text{video})}$ are individually fed into a video backbone network, and *K* corresponding hidden vectors are obtained. After the *K* hidden vectors are individually transformed by an MLP, they are aggregated into a single embedding vector using a pooling module Λ_p described in Section 4.3.

As the text and video backbone networks, we can choose state-of-the-art pre-trained models for texts and videos, respectively. An example of the text backbone network is Bidirectional Encoder Representations from Transformers (BERT) [20], which is a masked language model, while an example of the video backbone network is the SlowFast network for video recognition [21].

4.3. Pooling Modules

In the video set encoder, the pooling module Λ_p is used to aggregate *K* hidden vectors into a single vector, where *p* is the name of the pooling module. The pooling module must satisfy *the permutation invariant property* [22], so that the prediction over the video set encoder does not depend on the input order of the videos in $\mathbf{x}^{(video)}$. In addition, it is desirable that the number of hidden vectors *K* can change dynamically between training, validation, and test phases to cope with computational limitations and a lack of datasets. In our study, we consider three types of pooling modules that satisfy the above properties.

Suppose that we are given a set of *K* vectors $\{e_k\}_{k=1}^K$, where e_k is a d_{emb} -dimensional real-valued vector. The first and second pooling modules are *max pooling* $\Lambda_{max}(\cdot)$ and *mean pooling* $\Lambda_{mean}(\cdot)$, which output the maximum value and the mean value for each dimension over the set of vectors, respectively. The third one is *attentionbased pooling*, which calculates the weighted sum of the input vectors, where the weight for the *k*th vector, $a_k \in [0, 1]$, is determined based on a self-attention mechanism. This was originally proposed for multiple-instance learning [23]. Specifically, it is formalized as

$$\Lambda_{\text{att}}\left(\{\boldsymbol{e}_{k}\}_{k=1}^{K}\right) = \sum_{k=1}^{K} a_{k}\boldsymbol{e}_{k}, \quad a_{k} = \frac{\exp\left(\boldsymbol{U} \tanh(\boldsymbol{V}\boldsymbol{e}_{k}^{\top})\right)}{\sum_{k'=1}^{K} \exp\left(\boldsymbol{U} \tanh(\boldsymbol{V}\boldsymbol{e}_{k'}^{\top})\right)},$$
(5)

where $U \in \mathbb{R}^{1 \times d_{att}}$ and $V \in \mathbb{R}^{d_{att} \times d_{emb}}$ are parameters to estimate, and $tanh(\cdot)$ is the hyperbolic tangent function. It is expected that the more critical the vectors, the larger the weight.

4.4. Training

In training, we try to find optimal parameters Θ and Φ by minimizing a task-specific loss. For the detection task, we use a cross-entropy loss that consists of a term for positive samples and a term for negative samples. The term for positive samples, i.e., pairs of actions with a positive relation, is calculated as follows:

$$\mathbb{E}_{\mathcal{R}_{\text{train}}^{(\text{det})}}\left[-\log \tilde{y}_{ij}^{(m,\text{det})}\right] = -\frac{1}{\left|\mathcal{R}_{\text{train}}^{(\text{det})}\right|} \sum_{(i,j)\in\mathcal{R}_{\text{train}}^{(\text{det})}} \log \tilde{y}_{ij}^{(m,\text{det})}.$$
 (6)

Alternatively, the term for negative samples, i.e., pairs of unrelated actions, is calculated as follows:

$$\mathbb{E}_{\mathcal{U}_{\text{train}}^{(\text{det})}} \left[-\log(1 - \tilde{y}_{ij}^{(m,\text{det})}) \right] \\ = -\frac{1}{\left| \mathcal{U}_{\text{train}}^{(\text{det})} \right|} \sum_{(i,j) \in \mathcal{U}_{\text{train}}^{(\text{det})}} \log\left(1 - \tilde{y}_{ij}^{(m,\text{det})}\right), \tag{7}$$

where $\mathcal{U}_{\text{train}}^{(\text{det})} = \{(i, j) \mid i, j \in C_{\text{src}}, (i, j) \notin \mathcal{R}_{\text{train}}^{(\text{det})}\}$ is the pairs of unrelated actions. In total, the loss for the detection task is calculated as

$$\mathcal{L}^{(\text{det})}(\mathcal{D}_{\text{src}}^{(m)}, \mathcal{R}_{\text{train}}^{(\text{det})}; \Theta, \Phi)$$

$$= \mathbb{E}_{\mathcal{R}_{\text{train}}^{(\text{det})}} \left[-\log \tilde{y}_{ij}^{(m, \text{det})} \right] + \mathbb{E}_{\mathcal{U}_{\text{train}}^{(\text{det})}} \left[-\log(1 - \tilde{y}_{ij}^{(m, \text{det})}) \right].$$
(8)

Note that the size of $\mathcal{U}_{\text{train}}^{(\text{det})}$ is significantly greater than the size of $\mathcal{R}_{\text{train}}^{(\text{det})}$, and is close to $|C_{\text{src}}|^2$. Therefore, using all the samples in $\mathcal{U}_{\text{train}}^{(\text{det})}$ to compute the loss is computationally expensive. In practice, we approximate the loss (7) by drawing n_{neg} samples from $\mathcal{U}_{\text{train}}^{(\text{det})}$ uniformly at random, where n_{neg} is the number of negative samples to be determined in advance.

For the classification task, we use the cross-entropy loss between ground-truth relation label $y_{ij}^{(m,cls)}$ and the predicted one $\tilde{y}_{ij}^{(m,cls)}$ for actions *i*, *j* in modality *m*, which is defined as follows:

$$\mathcal{L}^{(\text{cls})}(\mathcal{D}_{\text{src}}^{(m)}, \mathcal{R}_{\text{train}}^{(\text{cls})}; \Theta, \Phi)$$

$$= -\frac{1}{\left|\mathcal{R}_{\text{train}}^{(\text{cls})}\right|} \sum_{(i,j, \mathbf{y}_{ij}^{(m,\text{cls})}) \in \mathcal{R}_{\text{train}}^{(\text{cls})}} \sum_{l=1}^{4} y_{ij\ell}^{(m,\text{cls})} \log \tilde{y}_{ij\ell}^{(m,\text{cls})},$$
(9)

where $y_{ij\ell}^{(m, \text{cls})}$ and $\tilde{y}_{ij\ell}^{(m, \text{cls})}$ indicate the values of the ℓ th elements of $y_{ij}^{(m, \text{cls})}$ and $\tilde{y}_{ij}^{(m, \text{cls})}$, respectively.

Minimizing each loss is performed by a stochastic gradient descent (SGD)-based optimization method. We describe the detailed implementation for training in Section 5.2.

4.5. Blending Predictions from Action Label and Video Set

Thus far, we have considered the model that takes only one modality as input, either action label texts or a set of videos. We now use both modalities simultaneously to make more accurate predictions. A simple but effective approach in this context is *blending*, which combines multiple predictions from different models [24]. The blending approach is a two-step process. First, we train a model for each of the two modalities individually, as described in Section 4.4. Then, we obtain the predictions in each modality on a validation set, and combine the predictions of the two modalities based on a logistic regression as follows:

$$\tilde{\boldsymbol{y}}_{ij}^{(\text{blend},t)} = \sigma^{(t)} \left(\boldsymbol{B}^{(t)} \text{concat} \left(\tilde{\boldsymbol{y}}_{ij}^{(\text{label},t)}, \tilde{\boldsymbol{y}}_{ij}^{(\text{video},t)} \right) + \boldsymbol{b}^{(t)} \right),$$
(10)

where, $\boldsymbol{B}^{(t)}$ is a blending parameter matrix such as $\boldsymbol{B}^{(\text{det})} \in \mathbb{R}^{1\times 2}$ and $\boldsymbol{B}^{(\text{cls})} \in \mathbb{R}^{4\times 2\cdot 4}$, and $\boldsymbol{b}^{(t)}$ is a bias parameter such as $\boldsymbol{b}^{(\text{det})} \in \mathbb{R}$ and $\boldsymbol{b}^{(\text{cls})} \in \mathbb{R}^{4}$. The parameters are trained on the validation set using the losses defined in (8) and (9) with $\mathcal{D}_{\text{val}}^{(m)}$ and $\mathcal{R}_{\text{val}}^{(t)}$ instead of $\mathcal{D}_{\text{train}}^{(m)}$ and $\mathcal{R}_{\text{train}}^{(t)}$.

5. Experiments

We conduct experiments to answer the three questions posed in Section 1 for the two tasks.

Table 1: Evaluation scores of action class relation detection on each target dataset. 'Label-only' and 'Video-only' denote our models that receive action labels and videos as input, respectively. 'Blending' denotes the blending approach described in Section 4.5, and 'Random' denotes the random guess approach. The bold typeface indicates the highest score on each target dataset.

	UCF101		HMDB51		ActivityNet		STAIR Actions		Charades		Kinetics-700	
	F1	AP	F1	AP	F1	AP	F1	AP	F1	AP	F1	AP
Label-only	0.650	0.711	0.587	0.530	0.628	0.635	0.595	0.558	0.406	0.293	0.578	0.570
Video-only	0.513	0.442	0.260	0.178	0.266	0.173	0.385	0.331	0.121	0.044	0.364	0.319
Blending	0.684	0.746	0.593	0.515	0.637	0.640	0.597	0.588	0.393	0.288	0.558	0.541
Random	0.013	0.006	0.013	0.006	0.012	0.006	0.012	0.006	0.004	0.002	0.009	0.005

Table 2: Accuracy of action class relation classification on each target dataset. The notation in this table is the same as Table 1.

	UCF 101	HMDB 51	Activity Net	STAIR Actions	Charades	Kinetics -700
Label-only	0.785	0.481	0.781	0.688	0.573	0.733
Video-only	0.715	0.367	0.591	0.560	0.373	0.621
Blending	0.794	0.398	0.690	0.686	0.598	0.738
Random	0.522	0.114	0.463	0.341	0.313	0.388

5.1. Evaluation Setting

To evaluate the performance of the proposed model, we split the six datasets in MetaVD into five source datasets and one target dataset, train our model described in Section 4 on the source datasets, and evaluate the performance in terms of the relation detection and classification between the source datasets and the target datasets. To investigate the performance changes due to the change in the target dataset, we create six variations of source and target datasets to assign each of the six datasets to the target dataset.

We evaluate our model for the detection task with F1 and average precision (AP) scores, which are commonly used in information retrieval [25]. For the classification task, we evaluate the model with a standard accuracy score. Higher F1, AP and accuracy scores are better.

5.2. Implementation Details

For the text backbone network, we use Sentence Transformers [26] with a publicly available pre-trained model called all-mpnet-base-v2, which outputs a 768dimensional continuous embedding vector for a text. The texts of the action labels are written in different writing styles, such as PascalCase, e.g., "BabyCrawling," and snake_case, e.g., "Getting_a_haircut." To get a good representation of the action labels, we convert the texts into normal phrases, such as "baby crawling" and "getting a haircut" before applying the text backbone network.

For the video backbone network, we use a pre-trained ResNet-101 SlowFast model [21] trained on Kinetics-400. By extracting the output of the penultimate layer of the ResNet-101 SlowFast model, we obtain a 2,304dimensional continuous embedding vector. Before applying the video set encoder, we extract a 32-frame video clip at a sampling rate of two by clipping the temporal middle of the video. We transform the RGB values of each video clip into continuous values from 0 to 1, and then, we standardize the values with a mean of 0.45 and a standard deviation of 0.225. We then resize the clip so that the size of its short side is 256. In the training phase, we apply spatially random cropping to 256×256 and horizontally random flipping to the clip. In the validation and test phases, we only center crop the clip to 256×256 . The video set that is input to the video set encoder is selected uniformly and randomly from all the videos associated with an action. The size of the set, K, is 10 for training and validation, and 30 for testing.

In both encoders, the MLP has multiple hidden layers and an output layer with batch normalization and ReLU activation, with all layers having d_{emb} units in common. Therefore, the outputs of both encoders, $h^{(label)}$ and $h^{(video)}$, are d_{emb} -dimensional vectors. In our experiments, we consistently set d_{emb} to 768. In the detection and classification heads, we use linear functions as $\mu^{(m,t)}(\cdot)$ for modality *m* and task t^1 .

In training, we optimize the parameters Θ and Φ by minimizing a task-specific loss. To approximate the

¹In our preliminary experiment, we tried to use MLPs as $\mu^{(m,t)}(\cdot)$, but the performance did not improve.



Figure 4: Accuracy comparison among pooling modules in action class relation classification. The accuracy of each pooling module is calculated by averaging test accuracies over different hyperparameters, and the error bar indicates the standard deviation.

loss (7), we set the number of negative samples n_{neg} to five times the number of positive samples $\mathcal{R}_{train}^{(det)}$. We use an Adam optimizer with an initial learning rate of 5e-4 and a batch size of 64. After five epochs, we change the learning rate to 5e-5. We terminate learning at 20 epochs. Note that we retain the original pre-trained parameters of the text and video backbone networks owing to the efficiency of the training.

The hyperparameters are the choice of the pooling modules described in Section 4.3 and the number of hidden layers in the MLP, which ranges from one to four. We choose the optimal hyperparameters that minimize the loss in the validation set.

5.3. Results

Tables 1 and 2 show the evaluation scores in action class relation detection and classification on each target dataset, respectively. To illustrate the difficulty of the tasks, we also show in these tables the scores of the random guess approach, which randomly predicts relation labels by following the label prior distribution in the training set.

The first question is: Can we build practical models for relation prediction using existing pre-trained neural network models for CV and natural language processing (NLP)? Compared to the random guess approach, all our approaches are much better at both detection and classification. The results indicate that the label and video set encoders can extract useful features from the action label texts and videos.

The second question is: Which modality of language and vision is better suited to predict the relations? We found that the label-only approach is consistently better than the video-only approach. We can hypothesize two reasons for this result. The first reason is that the relation label annotations in MetaVD were made from a linguistic perspective of the action label texts, rather than by viewing videos associated with the actions. The second reason is that the feature extraction is more difficult with the video set encoder than with the label encoder, as we will explain in Section 6.

The third question is: Can we improve prediction performance by using both modalities simultaneously rather than using only one? We found that the blending approach performs better than the label-only and video-only approaches in some cases, but becomes worse than the label-only approach in others. The blending approach optimizes its own parameters on a validation set constructed from the source datasets. Therefore, when the distribution of the validation set differs from that of the test set constructed from the target dataset, the blending approach would result in low scores.

Finally, in Fig. 4 we show the accuracy of each pooling module for action class relation classification. We found that the pooling module that achieves the highest accuracy varies depending on the target datasets. Furthermore, we found that the attention pooling tends to be unstable compared to the mean and max pooling because although it outperformed the others on ActivityNet and Kinetics-700, it proved to be the worst accuracy on the other target datasets. This result suggests that, although it is difficult to choose the best pooling module before training, from the perspective of stability, we should choose either the mean pooling or the max pooling.

5.4. Choice of Backbone Networks

In Section 5.3, we have shown the experiment results using Sentence Transformer with all-mpnet-base-v2 and ResNet-101 SlowFast as text and video backbone networks, respectively. In this subsection, we investigate how the predictive performances of the proposed model change when different types of backbone networks are used.

Table 3 shows the predictive performances of the labelonly and video-only approaches when different text and video backbone networks are used. As text backbone networks, we additionally use Sentence Transformer of sentence-t5-base [27] and LaBSE [28] models. The three text backbone networks have different network architectures and are trained with different datasets. We found that all-mpnet-base-v2 outperforms the others except in the classification accuracy for UCF-101. This result is identical to the benchmark results obtained by assessing the performances of various text embedding methods over 56 benchmark datasets [29]. As video backbone networks, we additionally use X3D-XS [30] and I3D [31] trained on Kinetics-400. We found that the predictive performances are better for ResNet-101 SlowFast, I3D, and X3D-XS in that order. The order is also identical to that of the predictive accuracy on the Kinetics-400 validation set². These results suggest that employing the models with higher benchmark scores as text and video backbone networks leads to better performances in action class relation detection and classification.

5.5. Analysis of Prediction Errors

We investigated the action class pairs that the proposed model incorrectly predicted. We found four major error types shown in Table 4. Error type (A) is caused by the miss-annotation of MetaVD. This indicates that the proposed model can find the miss-annotation of MetaVD, resulting in fixing the miss-annotation efficiently. Error type (B) contains action class pairs that can be interpreted as both related and unrelated. This error may be caused by the inconsistent annotations of similar relations in MetaVD. Error type (C) contains action class pairs that the model incorrectly judged unrelated even though their labels are similar. This error can be solved by exploiting string similarity between the action label texts. Error type (D) includes difficult cases to predict only from action label texts. This error can be improved by the blending approach. Indeed, the blending approach produced the predicted probabilities of 0.261 and 0.137

Table 3: The predictive performances with different text and video backbone networks, with (a) UCF101 and (b) Kinetics-700 as target datasets. The bold typeface indicates the highest score on each evaluation measure for label-only and video-only models, respectively.

(a) UC	CF101	Dete	ction	Classification	
	Backbone	F1	AP	Accuracy	
Label- only	all-mpnet-base-v2	0.650	0.711	0.785	
	sentence-t5-base	0.622	0.662	0.801	
	LaBSE	0.601	0.650	0.780	
Video- only	ResNet-101 SlowFast	0.513	0.442	0.715	
	I3D	0.404	0.311	0.598	
	X3D-XS	0.234	0.117	0.543	
(b) Kii	netics-700	Detection		Classification	
	Backbone	F1	AP	Accuracy	
Label- only	all-mpnet-base-v2	0.578	0.570	0.733	
	sentence-t5-base	0.472	0.431	0.677	
	LaBSE	0.489	0.481	0.719	
Video- only	ResNet-101 SlowFast	0.364	0.319	0.621	
	I3D	0.324	0.240	0.598	
	X3D-XS	0.074	0.027	0.543	

for the pair "Typing" and "using_computer" and the pair "walk" and "BandMarching", respectively, which are significantly higher than the predicted probabilities produced by the label-only approach.

6. Discussion

We have confirmed that our model achieves high performance in predicting relations in many cases. However, in the experiments, the following three difficulties were encountered, which should be addressed to further improve performance.

The first problem is *the class prior shift* between the source and target datasets, which occurs in the classification task and leads to a degradation of accuracy on the target dataset. For example, when the target dataset is Kinetics-700, its class prior distribution is [0.107, 0.605, 0.143, 0.143], while that of the source datasets is [0.129, 0.336, 0.267, 0.267]. This problem can be mitigated by introducing prior shift adaptation [32]. Along with the class prior shift, *the domain shift*, in which the distributions of inputs differ between the source and target datasets, can also occur because the target dataset,

²https://pytorchvideo.readthedocs.io/en/latest/ model_zoo.html

Error type	Action class in source dataset	Action class in target dataset	Related?	Predicted prob.
(A)	Wakeboarding (Kinetics-700)	Surfing (UCF101)	No	0.998
	sewing (STAIR Actions)	Knitting (UCF101)	No	0.998
(B)	longboarding (Kinetics-700)	SkateBoarding (UCF101)	No	0.995
	scuba_diving (Kinetics-700)	SkyDiving (UCF101)	No	0.992
(C)	Using_parallel_bars (ActivityNet)	ParallelBars (UCF101)	Yes	< 0.001
	making_pizza (Kinetics-700)	PizzaTossing (UCF101)	Yes	< 0.001
(D)	using_computer (STAIR Actions)	Typing (UCF101)	Yes	0.008
	walk (HMDB51)	BandMarching (UCF101)	Yes	0.002

Table 4: Four types of errors made by the label-only relation detection model. Examples of types (A) and (B) are unrelated action class pairs, whereas those for (C) and (D) are related according to the MetaVD annotations.

i.e., the external dataset to be augmented with MetaVD, is freely constructed by users. For this problem, unsupervised domain adaptation techniques [33] can be effective.

The second problem is to extract good features from the videos that represent an action. Although the videos in ActivityNet and Charades are relatively long, we used video clips of approximately 2 seconds by extracting the temporal middle of the videos according to the specification of the video backbone network we used. It is expected that by extracting the features from longer video clips, we can use richer information about the action for the predictions. For the same reason, it is also important to increase the size of the video sets, K.

The third problem is to improve the prediction performance by *multi-modal fusion*, i.e., learning a model with both action labels and videos as input in an endto-end manner. In our preliminary experiment, we have attempted an intermediate fusion strategy [34]. However, its performance was worse than that of the label-only approach. The result seems to be due to the negative effects of the difficulties mentioned above. Therefore, we believe that the performances can be improved by developing a multi-modal fusion model that is robust to these problems.

Through the error analysis in Section 5.5, we found that inconsistent annotations in MetaVD for the similar relations worsen the predictive performance of our relation prediction models. As stated in Section 1, we aim at exploiting our relation prediction models to augment users' own datasets using MetaVD. For this aim, we should consider treating similar relations as "unrelated" to avoid noisy dataset augmentation.

7. Conclusion

To augment an HAR dataset with MetaVD, the relationship between actions in the dataset and MetaVD needs to be inferred. We introduced a model that exploits two modalities, action label texts and video sets, to predict the relationship. With simulated experiments using one of the datasets in MetaVD as an imitated external dataset and inferring action relationship with the other datasets in MetaVD, we confirmed that: 1) the recent pre-trained neural networks in NLP and CV are effective; 2) the action label texts are more useful for predicting relations than the video sets; and 3) a blending approach that combines the predictions from both the modalities is superior to using only one of the modalities in some cases.

In our experiment, the result was evaluated by the accuracy of relation prediction, but the final goal is to train an HAR model on a real external dataset augmented by MetaVD with the relation prediction model. In future work, we will investigate the performance of the HAR model and how it is affected by the relation prediction accuracy.

Acknowledgments

This paper is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

[1] A. Tejero-de Pablos, Y. Nakashima, T. Sato, N. Yokoya, Human action recognition-based video summarization for RGB-D personal sports video, in: IEEE International Conference on Multimedia and Expo, 2016, pp. 1–6.

- [2] Y. Iinuma, S. Satoh, Video action retrieval using action recognition model, in: Proceedings of the 2021 International Conference on Multimedia Retrieval, 2021, pp. 603–606.
- [3] C.-B. Jin, S. Li, H. Kim, Real-time action detection in video surveillance using sub-action descriptor with multi-CNN, arXiv preprint (2017). arXiv: 1710.03383.
- [4] Y. Kong, Y. Fu, Human action recognition and prediction: A survey, arXiv preprint (2018). arXiv: 1806.11230.
- [5] Y. Yoshikawa, Y. Shigeto, A. Takeuchi, MetaVD: A meta video dataset for enhancing human action recognition datasets, Computer Vision and Image Understanding: CVIU 212 (2021) 103276.
- [6] K. Soomro, A. R. Zamir, M. Shah, UCF101: A dataset of 101 human actions classes from videos in the wild, arXiv preprint (2012). arXiv:1212.0402.
- [7] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, T. Serre, HMDB: A large video database for human motion recognition, in: Proceedings of the 2011 IEEE International Conference on Computer Vision, 2011, pp. 2556–2563.
- [8] F. Caba Heilbron, V. Escorcia, B. Ghanem, J. Carlos Niebles, Activitynet: A large-scale video benchmark for human activity understanding, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 961–970.
- [9] Y. Yoshikawa, J. Lin, A. Takeuchi, STAIR actions: A video dataset of everyday home actions, arXiv preprint (2018). arXiv:1804.04326.
- [10] G. A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, A. Gupta, Hollywood in homes: Crowdsourcing data collection for activity understanding, in: European Conference on Computer Vision, Lecture Notes in Computer Science 9905, Springer, 2016, pp. 510–526.

- [11] J. Carreira, E. Noland, C. Hillier, A. Zisserman, A short note on the kinetics-700 human action dataset, arXiv preprint (2019). arXiv:1907.06987.
- [12] S. Ji, S. Pan, E. Cambria, P. Marttinen, P. S. Yu, A survey on knowledge graphs: Representation, acquisition, and applications, IEEE Transactions on Neural Networks and Learning Systems 33 (2) (2022) 494–514.
- [13] R. Xie, Z. Liu, J. Jia, H. Luan, M. Sun, Representation learning of knowledge graphs with entity descriptions, in: Proceedings of the 30th AAAI Conference on Artificial Intelligence., 2016, pp. 2659– 2665.
- [14] R. Xie, Z. Liu, H. Luan, M. Sun, Image-embodied knowledge representation learning, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, 2017, pp. 3140–3146.
- [15] T. Hamaguchi, H. Oiwa, M. Shimbo, Y. Matsumoto, Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, 2017, pp. 1802–1808.
- [16] J. Cheng, L. Wang, J. Wu, X. Hu, G. Jeon, D. Tao, M. Zhou, Visual relationship detection: A survey, IEEE Transactions on Cybernetics 52 (8) (2022) 8453–8466.
- [17] H. Zhang, Z. Kyaw, S.-F. Chang, T.-S. Chua, Visual translation embedding network for visual relation detection, in: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5532–5540.
- [18] G. Gkioxari, R. Girshick, P. Dollar, K. He, Detecting and recognizing human-object interactions, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 8359–8367.
- [19] X. Shang, T. Ren, J. Guo, H. Zhang, T.-S. Chua, Video visual relation detection, in: Proceedings of the 25th ACM International Conference on Multimedia, 2017, pp. 1300–1308.

- [20] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North Americal Chapter of Association for Computational Linguistics, 2019, pp. 4171–4186.
- [21] C. Feichtenhofer, H. Fan, J. Malik, K. He, Slowfast networks for video recognition, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 6202–6211.
- [22] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. Salakhutdinov, A. Smola, Deep sets, arXiv preprint (2017). arXiv:1703.06114.
- [23] M. Ilse, J. Tomczak, M. Welling, Attention-based deep multiple instance learning, in: Proceedings of the 35th International Conference on Machine Learning, 2018, pp. 2127–2136.
- [24] D. H. Wolpert, Stacked generalization, Neural Networks: The Official Journal of the International Neural Network Society 5 (2) (1992) 241–259.
- [25] C. D. Manning, Introduction to Information Retrieval, Cambridge University Press, 2008.
- [26] N. Reimers, I. Gurevych, Sentence-BERT: Sentence embeddings using siamese BERT-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, 2019, pp. 3982–3992.
- [27] J. Ni, G. H. Ábrego, N. Constant, J. Ma, K. B. Hall, D. Cer, Y. Yang, Sentence-T5: Scalable sentence encoders from pre-trained text-to-text models, ArXiv preprint (2021). arXiv:2108.08877.
- [28] F. Feng, Y. Yang, D. Cer, N. Arivazhagan, W. Wang, Language-agnostic BERT sentence embedding, in: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2022, pp. 878–891.
- [29] N. Muennighoff, N. Tazi, L. Magne, N. Reimers, MTEB: Massive text embedding benchmark, arXiv preprint (2022). arXiv:2210.07316.

- [30] C. Feichtenhofer, X3d: Expanding architectures for efficient video recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 203–213.
- [31] J. Carreira, A. Zisserman, Quo vadis, action recognition? A new model and the kinetics dataset, ArXiv preprint (2017). arXiv:1705.07750.
- [32] T. Sipka, M. Sulc, J. Matas, The hitchhiker's guide to prior-shift adaptation, in: Proceedings of the 2022 IEEE/CVF Winter Conference on Applications of Computer Vision, 2022, pp. 1516–1524.
- [33] X. Liu, C. Yoo, F. Xing, H. Oh, G. El Fakhri, J.-W. Kang, J. Woo, Deep unsupervised domain adaptation: A review of recent advances and perspectives, arXiv preprint (2022). arXiv:2208.07422.
- [34] D. Ramachandram, G. W. Taylor, Deep multimodal learning: A survey on recent advances and trends, IEEE Signal Processing Magazine 34 (6) (2017) 96– 108.