Neural Networks 122 (2020) 1–23

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Multi-label zero-shot human action recognition via joint latent ranking embedding

Qian Wang, Ke Chen *

Department of Computer Science, The University of Manchester, UK

ARTICLE INFO

Article history:
Received 10 June 2019
Received in revised form 9 August 2019
Accepted 22 September 2019
Available online 21 October 2019

Keywords:
Human action recognition
Multi-label learning
Zero-shot learning
Joint latent ranking embedding
Weakly supervised learning

ABSTRACT

Human action recognition is one of the most challenging tasks in computer vision. Most of the existing works in human action recognition are limited to single-label classification. A real-world video stream, however, often contains multiple human actions. Such a video stream is usually annotated collectively with a set of relevant human action labels, which leads to a multi-label learning problem. Furthermore, there are a great number of meaningful human actions in reality but it would be extremely difficult, if not impossible, to collect/annotate sufficient video clips regarding all these human actions for training a supervised learning model. In this paper, we formulate a real-world human action recognition task as a multi-label zero-shot learning problem. To address this problem, a joint latent ranking embedding framework is proposed. Our framework holistically tackles the issue of unknown temporal boundaries between different actions within a video clip for multi-label learning and exploits the side information regarding the semantic relationship between different human actions for zero-shot learning. Specifically, our framework consists of two component neural networks for visual and semantic embedding respectively. Thus, multi-label zero-shot recognition is done by measuring relatedness scores of concerned action labels to a test video clip in the joint latent visual and semantic embedding spaces. We evaluate our framework in different settings, including a novel data split scheme designed especially for evaluating multi-label zero-shot learning. The experimental results on two weakly annotated multi-label human action datasets (i.e. *Breakfast* and *Charades*) demonstrate the effectiveness of our framework.

1. Introduction

As one of the most challenging tasks in computer vision, human action recognition refers to automatic recognizing human actions conveyed in a video clip. In last two decades, human action recognition has been extensively studied. As there are many different human actions in reality, this task is generally formulated as a multi-class classification problem. To train a multi-class classifier for human action recognition, a great number of examples for each single action are required in the current setting. To collect such training examples, one needs to manually trim a video stream to ensure that there is only one human action appearing in a trimmed video episode. This annotation process is laborious and time-consuming and there is hence no large-scale dataset with "fine-grained" annotation for human action recognition. In contrast to ImageNet (Deng et al., 2009) for object recognition, where it consists of a total of 3.2 million cleanly labelled images spreading over 5247 categories, there are much fewer annotated video clips involving only a small number of human actions. For instance, HMDB51 and UCF101 are among the most commonly used benchmark datasets in human action recognition, where there are 6676 and 13,320 instances of only 51 and 101 different human actions, respectively. The limitation of human action datasets in such a scale has become an obstacle in developing a large-scale human action recognition system.

In a real scenario, a video clip often conveys multiple human actions corresponding to different concepts. Hence, a set of multiple action labels have to be used to characterize its complete semantics underlying human actions conveyed in this video clip. For example, video clips on YouTube are usually uploaded by users along with some descriptive terms that can be used to infer the human actions conveyed in those video clips. In this circumstance, descriptive terms may be viewed as a set of coherent labels that collectively characterize the semantics at a global level. Recently, a very large multi-label video dataset YouTube-8M (Abu-El-Haija et al., 2016) has been collected by Google Research. Although the dataset is not restricted to human action video clips, it paves a new way for various video analyses including human action recognition. One of essential

* Corresponding author.
E-mail addresses: qian.wang173@hotmail.com (Q. Wang), ke.chen@manchester.ac.uk (K. Chen).

https://doi.org/10.1016/j.neunet.2019.09.029

video analysis problems on such a dataset may be formulated as multi-label learning that predicts a set of labels associated to a given instance or a set of relatedness scores corresponding to the candidate labels that could characterize this instance. In multi-label learning, a training example often consists of an input instance and a set of labels associated with this instance at a global level (no need of explicitly associating each of those labels to a relevant object within this instance). While multi-label data are common in many domains and multi-label classification has been studied under different applications (Zhang & Zhou, 2014), e.g., semantic image tagging, text categorization and gene functionality prediction, only few works are pertinent to multi-label human action recognition in literature due to a lack of human action datasets annotated with multiple class labels. To fill in this gap, a dataset dubbed Charades (Sigurdsson et al., 2016) was collected especially for multi-label human action recognition and made publicly available very recently. In addition, other datasets collected for different tasks were also considered to be used in multi-label human action recognition. Thus, such datasets provide a proper test bed for multi-label human action recognition studies.

Multi-label human action recognition often has to work on weakly labelled video data, i.e., the training data are annotated at the video level without exhaustively trimming and annotating multiple action episodes. While it is easier to collect such video clips associated with a set of labels at a global level than those with "fine-grained" annotation, it would be still very challenging to collect all the training examples due to the existence of many different human actions. *Zero-shot learning* (ZSL) provides an alternative solution to alleviate this problem. ZSL aims to recognize the instances belonging to novel classes which are not seen during training. It has been formally shown that under certain conditions, a ZSL system trained on a dataset of finite classes could be used to predict infinite number of classes unseen during the learning (Zhang, Acharyya, Liu, and Gong, 2016). Under the ZSL framework, we merely need to collect and annotate training examples for a moderate set of training classes and expect that a large number of novel classes can be recognized via exploiting the semantic relationship between different human actions. To this end, a ZSL algorithm needs to transfer the knowledge regarding the relations between visual features and class label semantics learned from known or training classes to unseen or test classes. The knowledge transfer is enabled by modelling the *semantic representations* of different classes, which can be easily obtained from side information, e.g., descriptive texts, with a much less effort than collecting and annotating visual data. Nevertheless, most of the existing ZSL methods were proposed to tackle single-label ZSL problems but multi-label ZSL problems are much more complicated, leading to additional challenge that do not exist in single-label ZSL. Although some of single-label ZSL methods might be extensible to multi-label scenarios, their effectiveness of different ZSL algorithms have not been extensively investigated in the multi-label learning scenarios. To the best of our knowledge, there exists no work in multi-label zero-shot human action recognition.

In this paper, we address the multi-label ZSL issues in the context of human action recognition. In our problem, the training video data are weakly labelled so that the exact temporal–spatial locations of multiple human actions in a video clip remain unknown. In addition, training examples consisting of visual instances and their corresponding label sets of multiple labels are only available for those associated with training/known labels, a subset of the action label collection considered in the recognition stage. Thus, the nature of multi-label zero-shot human action recognition poses several challenges that do not exist in static data and single-label ZSL. To tackle all the challenges in a holistic way, we propose a novel *joint latent ranking embedding* framework. The framework aims to learn a joint latent ranking embedding from visual and semantic domains. By using the learned joint latent ranking embedding, any visual instances and any action labels can be mapped into the joint latent visual and semantic embedding spaces where positive connections between visual instances and action labels rank ahead of negative ones. Thus, any human actions can be recognized regardless of known or unseen actions during learning. Our framework consists of two component models: *visual* and *semantic* models. The visual model learns mapping a visual instance into the latent *visual embedding* space, while the semantic model learns mapping action labels into the latent *semantic embedding* space. The visual and the semantic models are tightly coupled to learn a proper ranking that works in the joint latent visual and semantic embedding spaces with an alternate learning algorithm on training examples annotated with only known action labels. In the test, multi-label zero-shot recognition is done by measuring relatedness scores of action labels to a test visual clip in the joint latent visual and semantic embedding spaces.

Our main contributions in this paper are summarized as follows:

- By considering real scenarios, we formulate general human action recognition as a *multi-label zero-shot learning* problem . To the best of our knowledge, our work presented in this paper is among the first attempts in studying human action recognition from a multi-label zero-shot learning perspective, which tackles several technical challenges pertaining to this problem in a holistic way.
- To address the multi-label zero-shot issues arising from weakly annotated data for human action recognition, we propose a novel joint latent ranking embedding framework consisting of visual and semantic embedding models. To train two embedding models effectively, we come up with a learning algorithm that alternately optimizes the parameters in two embedding models via minimizing the proper rank loss functions.
- To test the performance of our proposed framework, we conduct a thorough evaluation via a comparative study on two benchmark multi-label human action datasets, *Breakfast* and *Charades*, with various evaluation metrics and different settings including a novel data split protocol simulating a real scenario of multi-label zero-shot human action recognition.

The rest of this paper is organized as follows. Section 2 reviews related works. Section 3 presents our framework for multi-label zero-shot human action recognition. Section 4 describes our experimental settings, and Section 5 reports the experimental results. The last section draws conclusions.

2. Related work

In this section, we review the existing works relating to multi-label human action recognition, especially for those applicable to multi-label zero-shot learning scenarios, and point out the limitations of the existing works. We first overview existing multi-label classification methods and then focus on the existing works in multi-label ZSL learning despite the fact that none of such multi-label ZSL methods has been applied to human action recognition. Finally, semantic representations required by any ZSL methods are briefly reviewed.

2.1. Multi-label learning

In a real scenario, semantics underlying real-world data is often complex and has to be characterized with multiple labels, e.g., web videos. In a video clip pertaining to human actions, multiple actions could happen simultaneously, e.g., sitting, eating and listening. In this scenario, no episode in such a video clip can be characterized by a single action label and a set of labels hence have to be used collectively to describe this video clip. Even though a video clip can be divided into several episodes corresponding to different human actions, the segmentation and annotation process could be difficult, tedious, laborious and time-consuming. In particular, semantic image segmentation and human action detection in video streams remain unsolved in general. As a result, multi-label learning is often formulated as a weakly supervised learning task that predicts a set of labels associated with an instance but does not address the issue in assigning each label in the set to a specific object within this instance. To tackle a weakly supervised multi-label learning problem, two different representation methods are used to characterize input data: *instance-level* and *object-level* representations. An instance-level representation is a global representation of an instance, e.g., a video clip or an image, without considering objects appearing in this instance, while an object-level representation is a local representation that describes individual objects extracted from an instance, e.g., semantically meaningful episodes/patches in the video/image. Depending on the representation of input data, multi-label learning methods can be divided into two categories.

In multi-label learning, most of the existing methods (Guillaumin, Mensink, Verbeek, & Schmid, 2009; Nam, Kim, Mencía, Gurevych, & Fürnkranz, 2014; Wang, Jia, and Breckon, 2019; Wang et al., 2016; Zhang, Gong, and Shah, 2016) work on an instance-based representation, a single feature vector of an instance. Recently, Fast0Tag (Zhang, Gong, and Shah, 2016) was proposed for multi-label image tagging by learning a mapping from visual to label space. An image containing multiple objects is represented by one aggregated visual representation. Alternatively, TagProp (Guillaumin et al., 2009) uses an adapted nearest neighbour model for multi-label learning in visual space where each image of multiple objects is also represented by one feature vector at the instance level. Wang et al. (2016) use a convolutional neural network (CNN) directly working on raw images of multiple objects to learn image-level deep visual representations for multi-label classification. Nam et al. (2014) use a deep neural network with a rank loss in learning for large-scale multi-label text classification where an input document is represented with a single feature vector. Although representing one instance at the global level is straightforward and convenient, it might neglect the intrinsic relationship between multiple objects within an instance. Thus, an instance-level representation might result in a catastrophic information loss, especially for long-term dependent and complex video data.

To overcome the weakness in neglecting the information regarding the intrinsic relationship between objects within an instance, efforts have been made to exploit such information in previous works. Despite being difficult, the segmentation of multiple objects within an instance turns out to be beneficial to multi-label learning. One framework named *multi-instance multi-label learning (MIML)* (Zhou & Zhang, 2007) demonstrates that multi-label learning can be fulfilled effectively if multiple objects within an instance have been explicitly separated or segmented even if no label is explicitly assigned to each of multiple objects within an instance during learning. In real applications, however, automatic semantic segmentation of objects in an instance is also challenging, and a manual segmentation process is laborious and time-consuming. Moreover, some recent works tend to explore object-based representations without using any explicit semantic object segmentation techniques, which seeks a synergy between the MIML and object-level representations. Gu et al. (2016) address this weakly supervised issue in multi-label human action detection with a two-stage solution. First, a set of potential objects or spatial–temporal volumes are generated and selected from a video instance with a set of handcrafted rules. Then the problem is transformed into a MIML problem which can be solved by those traditional multi-label learning algorithms under the MIML framework. Similar ideas were also explored by Tang, Wang, Huang, Bai, and Liu (2017) and Wei et al. (2016) for multi-label image classification. However, the extraction of true positive objects from the original visual instance is a very challenging yet non-trivial task, which critically determines the multi-label learning performance. To extract all the meaningful objects within an instance, a lot of candidate proposals have to be considered so that it might suffer from a high computational burden. Instead of using the MIML, Cabral, De la Torre, Costeira, and Bernardino (2015) attempt to explore the information regarding multiple objects in instances via a matrix completion method. Their method works on the assumption that an instance representation may be expressed by a linear combination of hidden representations of objects appearing in this instance. Experimental results reported by Cabral et al. (2015) demonstrate the effectiveness of this method via an instance-level bag-of-words image representation. However, this idea does not seem applicable to other kinds of visual representations, such as those popular yet powerful deep representations.

2.2. Multi-label zero-shot learning

Zero-shot learning (ZSL) has attracted much attention in recent years and provides a promising technique for recognizing a large number of classes without the need of the training data concerning all the classes. Very recently, Zhang, Acharyya, Liu, and Gong (2016) have formally shown that it is feasible to predict a collection of infinite unseen labels with a classifier learned on training data concerning only a number of labels in this collection or a subset of this collection, where multi-label ZSL is a special case in this so-called "infinite-label learning" paradigm. According to a ZSL taxonomy (Wang & Chen, 2017b), existing ZSL approaches are divided into three categories, namely, *direct mapping* (Akata, Reed, Walter, Lee, & Schiele, 2015; Fu & Huang, 2010; Lampert, Nickisch, & Harmeling, 2014; Xian et al., 2016; Yu, Ji, Guo, and Pang, 2018), *model parameter transfer* (Changpinyo, Chao, Gong, & Sha, 2016; Mensink, Gavves, & Snoek, 2014) and *joint latent space learning* (Changpinyo, Chao, & Sha, 2017; Frome et al., 2013; Lei Ba, Swersky, Fidler, et al., 2015; Wang & Chen, 2017b; Yu, Ji, Guo, and Zhang, 2018; Zhang & Saligrama, 2015, 2016; Zhang, Xiang, & Gong, 2017). More recently, *synthetic feature generation* methods based on general adversarial networks (GANs) (Goodfellow et al., 2014) have become prevalent and achieved state-of-the-art performance in zero-shot learning (Huang, Wang, Yu, & Wang, 2019; Xian, Lorenz, Schiele, & Akata, 2018). Although most existing works focus on single-label ZSL, efforts have been made to address more complex multi-label ZSL issues (Fu, Yang, Hospedales, Xiang, & Gong, 2014; Lee, Fang, Yeh, & Frank Wang, 2018; Mensink et al., 2014; Nam, Mencía, Kim, & Fürnkranz, 2015; Ren, Jin, Lin, Fang, & Yuille, 2017; Zhang, Gong, and Shah, 2016).

For *direct mapping*, it needs to learn a mapping directly from visual to semantic space for zero-shot recognition on the semantic space, which poses a challenge to multi-label ZSL. In single-label ZSL, a training example provides a visual-semantic representation pair used to learn a one-to-one direct mapping. In multi-label ZSL, however, one instance has to be associated

with a set of multiple labels and the number of labels associated with different instances are various. As a label is represented with a semantic feature vector, e.g., a vector of attributes or a word vector, in a semantic space, it is no longer straightforward to learn a direct mapping from visual to semantic space in the context of multi-label ZSL. How to model complex semantics underlying a set of labels associated with an instance becomes a central issue in multi-label ZSL. To tackle this issue, most of existing works (Fu et al., 2014; Sandouk & Chen, 2016b) make use of the composition properties of semantic representations such as word vectors by using the average of semantic representations of multiple labels to a collective semantic representation for a set of labels associated with the instance. Thus, a training example is formed with a pair of an instance-level visual representation and its corresponding collective semantic representation, which enables one to learning a direct mapping for multi-label ZSL. Apparently, such a collective representation cannot avoid information loss even though a contextualized semantic representation (Sandouk & Chen, 2016b) was used. In particular, the multi-label ZSL method proposed by Fu et al. (2014) is subject to a fundamental limitation; their method has to take into account all the possible combinations of different unseen labels in a pre-fixed unseen label collection. Thus, the computational complexity of their algorithm grows exponentially with respect to the number of unseen labels and hence can cope with only a very small number of pre-fixed unseen labels (e.g., up to eight in their experiments). To alleviate the information loss problem in generating a collective semantic representation, Fast0Tag (Zhang, Gong, and Shah, 2016) introduces an alternative solution to collective semantic representations. In their method, each visual instance is mapped into a "principal direction" in the semantic space based on an assumption that there is always such a direction for any multi-labelled instances in a semantic space, e.g., word vector space, and all the labels associated with this instance always rank ahead of irrelevant labels. In other words, a hyperplane perpendicular to this direction can always be found to separate the relevant labels from the irrelevant ones for any multi-labelled instance. While this assumption holds for those datasets used in their zero-shot image tagging experiments (Zhang, Gong, and Shah, 2016), it remains unclear for other image datasets and different domains, e.g., human action recognition. From an alternative perspective, Ren et al. (2017) suggest using an object-level visual presentation under the direct mapping framework for multi-label zero-shot object recognition. Before multi-label learning takes place, an image thus has to be semantically segmented into meaningful subregions and each subregion can be characterized by one label. As a result, their solution is actually a special case of the MIML (Zhou & Zhang, 2007) but heavily relies on sophisticated semantic segmentation techniques that remain unavailable up to date. Furthermore, this method is not extensible to sequential data such as video clips.

Like the works in extending *direct mapping* to multi-label ZSL, the *model parameter transfer* idea is also adapted for multi-label ZSL, leading to COSTA (Mensink et al., 2014). COSTA aims to establish a model for each unseen label via a linear weighted combination of known-label models. The known-label models are trained independently by means of a one-vs-rest binary classifier, e.g., support vector machines (SVMs). The combination coefficients are determined by the co-occurrence of multiple labels derived from either annotation of datasets in hand or external web sources. In COSTA, however, the known-label models are trained independently without considering the relationship and coherence among those labels that together describe an instance. Then, COSTA only uses label co-occurrences to model the relatedness between a pair of labels but neglects the semantics of an individual label itself. So far, this idea has been tested only on static images in the context of multi-label zero-shot object recognition.

The *joint latent space learning* methodology was proposed for multimedia information retrieval and multi-label related learning and led to favourable results in real-world applications (Gong, Ke, Isard, & Lazebnik, 2014; Karpathy & Fei-Fei, 2015; Weston, Bengio, & Usunier, 2010). The core idea underlying this methodology is learning a joint latent embedding from both visual and semantic domains to narrow the semantic gap so that a task can be done effectively in the latent embedding space(s). More recently, this general idea has also been explored in single-label ZSL (Changpinyo et al., 2017; Frome et al., 2013; Lei Ba et al., 2015; Wang & Chen, 2017b; Yu, Ji, Guo, and Zhang, 2018; Zhang & Saligrama, 2015, 2016; Zhang et al., 2017). Empirical studies suggest that those *joint latent space learning* methods often outperform most of existing direct mapping and model parameter transfer methods on several benchmark datasets designed for single-label ZSL (Changpinyo et al., 2017; Frome et al., 2013; Lei Ba et al., 2015; Wang & Chen, 2017b; Xian, Schiele, & Akata, 2017; Zhang & Saligrama, 2015, 2016; Zhang et al., 2017). In this paper, we propose a novel approach to multi-label zero-shot human action recognition by exploring the joint latent space learning idea to holistically tackle those challenges described in Section 1.

2.3. Semantic representation

Regardless of different ZSL scenarios, modelling semantics underlying a collection of labels and their relatedness plays a critical role in knowledge transfer required by ZSL. Miscellaneous methods in semantics modelling and representations have been developed from different perspectives including *attributes of labels*, *label embedding*, *co-occurrence of labels* and *concept embedding*.

Attributes of labels are a generic semantic representation where a label is characterized by a list of attributes common to all the labels (Lampert et al., 2014). Label embedding refers to embedding labels onto a semantic space where the semantic relatedness of labels are modelled (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013). Label embedding is often carried out via learning on external textural resources. For example, the famous *Word2Vec* semantic embedding is obtained by training a skip-gram neural network on the large-scale corpora, e.g., Google News dataset (Mikolov et al., 2013). Such semantic representations are widely used in ZSL (Fu et al., 2014; Ren et al., 2017; Wang & Chen, 2017b; Zhang, Gong, and Shah, 2016). Unlike label embedding obtained with external resources , co-occurrence of labels is yet another way to capture the relatedness between different labels (e.g., Nam et al., 2015). Alternatively, the co-occurrence information on different class labels can also be extracted from external resources for ZSL (Mensink et al., 2014). In particular, co-occurrence of labels allows for capturing the relatedness between labels jointly used to describe an instance. The label co-occurrence information may be incorporated into learning semantic embedding for a given dataset (e.g., Nam et al., 2015). In concept embedding, the semantic meaning of a label is assumed to be polysemous depending on different labels (together treated its context of this target label) jointly used to describe an instance. Hence, the semantic meaning of a label under a specific context frames a concept. As a result, concept embedding (Sandouk & Chen, 2016a) can be viewed as contextualized label embedding where a label may have multiple semantic representations in different contexts. The concept embedding seems specific and is only applicable to direct mapping for multi-label ZSL (Sandouk & Chen, 2016b). Our proposed framework for multi-label zero-shot human action is generic so that all the semantic representations apart from the concept embedding may be used directly in our framework.

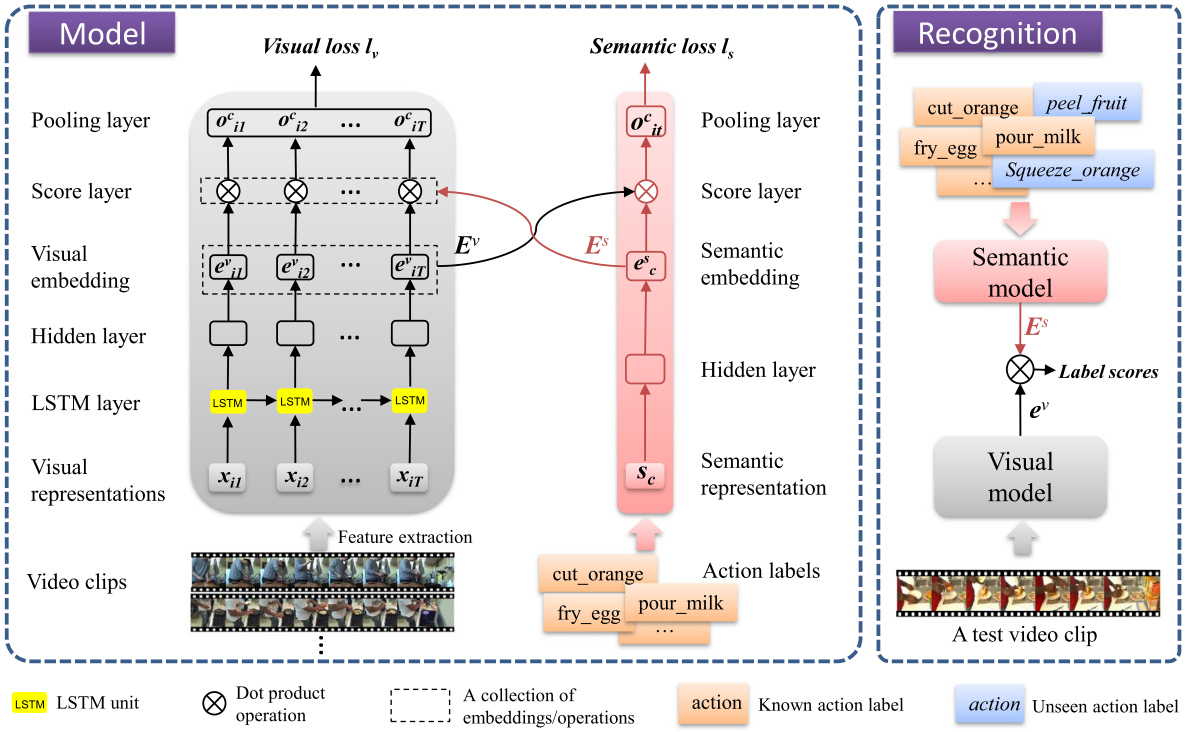

Fig. 1. Our multi-label zero-shot human action recognition framework. This framework shown in the left box is composed of two component models: visual and semantic model highlighted with grey and red colours, and both models are trained jointly with an alternative learning algorithm for the joint latent ranking embedding learning (c.f. Section 3.2). After the learning is completed, the trained visual and semantic models work together for multi-label zero-shot recognition via ranking the relatedness of known and unseen action labels to a test video clip (c.f. Section 3.3), as shown in the right box. Action labels marked with brown colour are training classes or known labels during learning, while action labels marked with blue colour are test classes or unseen labels during learning. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

3. Model description

In this section, we present a novel framework for multi-label zero-shot human action recognition. First, we present an overview of the proposed framework along with our motivation and justification. Next, we present the joint latent ranking embedding learning method including the rank loss functions and an alternate learning algorithm especially developed for our proposed architecture. Finally, we specify a procedure on how to apply a trained joint latent ranking embedding model to multi-label zero-shot human action recognition in test.

3.1. Overview

Our proposed framework aims at multi-label zero-shot human action recognition. We formulate this problem as learning a mapping $\boldsymbol{\phi} : \boldsymbol{x} \rightarrow \boldsymbol{y}$, where $\boldsymbol{x}$ is a visual input, e.g., a set of segment-level visual feature vectors extracted from a video clip, and $\boldsymbol{y} \in \mathbb{R}^{|C|}$ is a list of label-relatedness scores for $\boldsymbol{x}$ with respect to a action label collection, $C = \{1, \ldots, |C|\}$, where C is further divided into two mutually exclusive label subsets, C^{Tr} and C^{U}, corresponding to known (training) and unseen actions; i.e., $C^{Tr} \cup C^{U} = C$ and $C^{Tr} \cap C^{U} = \emptyset$. During learning the mapping $\boldsymbol{\phi}$, only training examples of labels in C^{Tr} are available. However, the learned mapping $\boldsymbol{\phi}$ is used to predict any actions appearing in a video clip no matter whether they are known actions in C^{Tr} or unseen actions in C^{U}.

To tackle the problem formulated above, we propose a joint latent ranking embedding framework. Motivated by the joint latent space learning idea used in ZSL (Frome et al., 2013; Lei Ba et al., 2015; Wang & Chen, 2017b; Zhang & Saligrama, 2015, 2016; Zhang et al., 2017), we would tackle the knowledge transfer issue in the joint latent embedding spaces where the original visual and semantic representations are mapped into. By embedding visual and semantic representations into the joint latent embedding spaces, we expect that semantic gap can be narrowed considerably and the semantic relatedness of known and unseen labels may be effectively explored and exploited in zero-shot recognition. Thus, our framework consists of two component models: *visual* and *semantic* models used to learn latent visual and semantic embedding, respectively. Two component models are tightly coupled to learn a joint latent ranking embedding for knowledge transfer, as illustrated in the left box of Fig. 1.

For visual embedding, we encounter two major technical issues due to the nature of weakly annotated data: (a) for a visual input, it remains unknown where an episode conveying an action, and (b) it remains unclear which of those action labels describing a video clip is associated with a specific video episode. Nevertheless, a video clip is an ordered sequence of frames and we could explore the temporal coherence underlying a video clip to tackle two aforementioned technical issues. Motivated by recent works in video classification and activity recognition (Donahue et al., 2015; Ma, Chen, Kira, & AlRegib, 2019; Yue-Hei Ng et al., 2015), we employ a *long short-term memory* (LSTM) (Hochreiter & Schmidhuber, 1997) recurrent neural network layer to capture temporal coherence underlying an action episode. Thus, the LSTM layer is first used to process a sequence of visual representations extracted from video segments. With the memorizing and forgetting mechanism of LSTM units, we expect that the LSTM layer explores the temporal structure of human actions conveyed in a video sequence (Wang, Yuan, and Wang, 2019); the LSTM units would memorize the input segments until parsing an episode regarding a human action is completed and then forget all the previous input segments when an episode conveying another

Table 1
Nomenclature.

Notation	Description
$\mathcal{D}$, $\mathcal{D}_T$	Training, test datasets
$\|\cdot\|$, $\|\|\cdot\|\|_1$	Cardinality of a set, L_1 norm of a vector
C, C^{Tr}, C^U	Label collection, training and unseen class label subsets
E^v, E^s	Visual and semantic embedding space
$\boldsymbol{x}_{it}$	Visual representation for the tth segment of the ith example
$\boldsymbol{x}_i$	Collection of all the segment-level visual representations of the ith example
$\boldsymbol{s}_c$	Semantic representation for the cth label
$\boldsymbol{Y}$	Binary target label matrix of training dataset
$\boldsymbol{y}_i$	Binary target label set of the ith example, i.e., the ith column of matrix $\boldsymbol{Y}$
$\boldsymbol{y}^c$	Binary indicator vector of the cth label appearing examples, i.e., the cth row of matrix $\boldsymbol{Y}$
$\boldsymbol{E}^v$, $\boldsymbol{e}_i^v$	Visual embedding matrix and the column vector for ith video clip
$\boldsymbol{E}^s$, $\boldsymbol{e}_c^s$	Semantic embedding matrix and the column vector for the cth label
d_x, d_s, d_e	Dimensions of visual, semantic, latent embedding space
$\boldsymbol{o}_i$	Relatedness scores between the ith example and all the candidate labels in visual model
$\boldsymbol{o}^c$	Relatedness scores between the cth label and all the training examples in semantic model
$\boldsymbol{\phi}_v$, Θ_v	Visual embedding function and parameters
$\boldsymbol{\phi}_s$, Θ_s	Semantic embedding function and parameters
$\boldsymbol{\phi}$	$\boldsymbol{\phi} = \{\boldsymbol{\phi}_v, \boldsymbol{\phi}_s\}$, mapping function for multi-label zero-shot recognition
$C(\hat{\boldsymbol{x}})$, $L(\hat{\boldsymbol{x}})$	The ground-truth label set of test instance $\hat{\boldsymbol{x}}$, the ranking list of all the labels predicted for $\hat{\boldsymbol{x}}$ in terms of the relatedness scores
D^c, L^c	Set of test instances of which ground-truth label sets include the cth label, the ranking list of all the test instances in terms of the relatedness scores for the cth label

action starts. Thus, an implicit saliency detection is carried out where no action episode boundaries are explicitly specified. For visual embedding, we further employ two fully-connected layers, dense layer of *rectified linear* (ReLu) units (Nair & Hinton, 2010) and visual embedding layer of linear units, to capture salient features on the temporal coherence representations yielded by the LSTM layer. While this specific visual model shown in the left box of Fig. 1 is used in our experiments, its capacity can be increased by adding more hidden units and/or layers if necessary. The score and average pooling layers above the visual embedding layer are used for joint latent ranking embedding learning as presented in Section 3.2. Thus, the visual model is carried out by a deep network of heterogeneous layers.

For semantic embedding, we employ a three-layer fully-connected neural network, the input layer, the hidden layer of ReLu units and the semantic embedding layer of linear units, to carry out the semantic model, as shown in the left box of Fig. 1. This learning model is capable of capturing the intricate semantic relatedness between different actions in a label collection of a moderate size, e.g. those datasets used in our experiments. If necessary, its capacity can be increased by adding more hidden units and/or layers. As a result, the neural network is fed with a specific semantic representation of action labels, e.g., word vectors and subsequently map them into the semantic embedding layer via a hidden layer. Likewise, the score and average pooling layers above the semantic embedding layer are used for joint latent ranking embedding learning. To explore the semantic relatedness between different labels in bridging the semantic gap between visual and semantic space, semantic embedding learning needs to automatically exploit the information carried in training data, e.g., frequency of label co-occurrence in a training dataset.

During the joint latent ranking embedding learning, the visual and semantic models are tightly coupled to learn a ranking criterion for the joint latent visual and semantic embedding spaces. This ranking criterion ensures that the relatedness scores of those labels associated with a visual instance are higher than those for other labels irrelevant to this instance, and the relatedness scores of those visual instances relevant to an action label are higher than those of other visual instances irrelevant to this label. For learning, we propose an algorithm working alternately on two models for parameter estimation by promoting the correct ranking based on training examples in known classes. During training, the visual model learns the visual embedding of a video instance such that those labels relevant to this instance rank ahead of other irrelevant ones in terms of the relatedness scores estimated on the semantic latent space, E^s. Reciprocally, the semantic model learns the semantic embedding of action labels such that the relevant visual instances rank higher than the irrelevant ones in terms of relatedness scores calculated in the visual latent space, E^v. Once the learning is completed, the trained joint latent ranking embedding model can be applied to a test video clip for human action recognition. As a result, the relatedness scores corresponding to all the known and unseen action labels in a label collection are achieved by using both visual and semantic models (c.f. Section 3.3), as illustrated in the right box of Fig. 1.

3.2. Joint latent ranking embedding learning

Now we present the joint latent ranking embedding learning in our proposed framework. To facilitate our presentation, we summarize the notations used in this paper in Table 1.

3.2.1. General description

Given a training set of weakly annotated video clips, $\mathcal{D} = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{|\mathcal{D}|}$, where $\boldsymbol{x}_i$ is the visual input and $\boldsymbol{y}_i \in \{+1, -1\}^{|C^{Tr}|}$ is its binary target label vector in the ith example: $+1/-1$ element indicates the presence/absence of a specific action belonging to C^{Tr} in $\boldsymbol{x}_i$.

For a video instance $\boldsymbol{x}_i$ in $\mathcal{D}$, we divide it evenly into T segments,[1] segment-level visual representations are extracted, which are collectively denoted by $\{\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \ldots, \boldsymbol{x}_{iT}\}$. At the tth time step, the segment representation $\boldsymbol{x}_{it}$ is fed into a hidden LSTM layer and processed by this LSTM layer and two subsequent fully-connected layers of linear activation functions (c.f. Fig. 1). The latent visual embedding of the tth segment, $\boldsymbol{e}_{it}^v$, is obtained as follows:

$$\boldsymbol{e}_{it}^v = \boldsymbol{\phi}_v(\boldsymbol{x}_{it}; \Theta_v). \tag{1}$$

Here, $\boldsymbol{\phi}_v$ is the visual embedding function implemented by the parametric visual model and Θ_v is a collective notation of all the parameters in this model, including weights and biases involved in this deep network.

Likewise, as depicted in Fig. 1, the cth label in a label collection is first represented by a specific semantic representation, $\boldsymbol{s}_c$, that is fed to the semantic embedding function, $\boldsymbol{\phi}_s$, implemented by the parametric semantic model of which all the parameters are denoted by Θ_s collectively. Thus, the semantic embedding, $\boldsymbol{e}_c^s$, of the cth label is

$$\boldsymbol{e}_c^s = \boldsymbol{\phi}_s(\boldsymbol{s}_c; \Theta_s). \tag{2}$$

A score layer is employed in each of the visual and the semantic models. In the visual model, the score layer takes the outputs of the visual embedding layer at all the time steps to yield the relatedness scores regarding all the labels for $\boldsymbol{x}_i$ with a dot product between the visual embedding of each segment in $\boldsymbol{x}_i$ and the semantic embedding of all the labels in a label collection:

$$\boldsymbol{o}_{it} = \langle \boldsymbol{e}_{it}^v, \boldsymbol{E}^s \rangle, \tag{3}$$

[1] A segment refers to a volume of multiple consecutive frames.

where $\boldsymbol{E}^s \in \mathbb{R}^{d_e \times |C^{Tr}|}$ is a collective notation of the semantic embedding of all the labels. Here, $\langle \boldsymbol{a}, \boldsymbol{B} \rangle = \boldsymbol{a}^T \boldsymbol{B}$ is a vectorial notation of the dot product between a vector, $\boldsymbol{a}$, and each column of a matrix, $\boldsymbol{B}$. Then the relatedness scores between this video instance and different labels are achieved by averaging over the scores on all the segments of this video instance:

$$\boldsymbol{o}_i = \frac{1}{T} \sum_{t=1}^{T} \boldsymbol{o}_{it}. \tag{4}$$

Likewise, the relatedness scores between different video instances and the cth label in the label collection, $\boldsymbol{o}^c \in \mathbb{R}^{|\mathcal{D}| \times 1}$, is estimated in the same manner as done in the visual model based on the visual embedding of those video instances and the semantic embedding of the cth label. Thus, the ith element of $\boldsymbol{o}^c$, the relatedness score regarding the ith visual instance is

$$o_i^c = \frac{1}{T} \sum_{t=1}^{T} \langle \boldsymbol{e}_{it}^v, \boldsymbol{e}_c^s \rangle. \tag{5}$$

For the joint latent ranking embedding learning, we need to optimize the parameters in the visual and the semantic models with training data and proper rank loss functions (the technical details are presented in Section 3.2.2). Assume that $l_v(\cdot, \cdot)$ and $l_s(\cdot, \cdot)$ are two loss functions with respect to the visual and the semantic model, the joint latent ranking embedding learning is boiled down to simultaneously solving the following optimization problems:

$$\Theta_v^* = \text{argmin}_{\Theta_v} \sum_{i=1}^{|\mathcal{D}|} l_v(\boldsymbol{o}_i, \boldsymbol{y}_i), \tag{6}$$

$$\Theta_s^* = \text{argmin}_{\Theta_s} \sum_{c \in C^{Tr}} l_s(\boldsymbol{o}^c, \boldsymbol{y}^c). \tag{7}$$

Here, the binary indicator vector $\boldsymbol{y}^c \in \mathbb{I}^{1 \times |\mathcal{D}|}$ is a row vector in the target label matrix $\boldsymbol{Y} \in \mathbb{I}^{|C^{Tr}| \times |\mathcal{D}|}$ of a training dataset, $\mathcal{D}$, and elements of $+1$ in $\boldsymbol{y}^c$ indicate that the cth label appears in the target label sets of the corresponding training examples in $\mathcal{D}$. The binary indicator vector $\boldsymbol{y}_i \in \mathbb{I}^{|C^{Tr}| \times 1}$ is a column vector in $\boldsymbol{Y}$, and elements of $+1$ in $\boldsymbol{y}_i$ refer to those labels in the target label set associated with the ith training example in $\mathcal{D}$. The value of elements corresponding to irrelevant visual input in $\boldsymbol{y}^c$ or labels in $\boldsymbol{y}_i$ is always set to -1.

3.2.2. Rank loss function

As described in Section 3.1, multi-label zero-shot learning needs to establish a mapping that outputs a label-relatedness score list for a video input where the scores of the relevant labels should be ranked higher than those of irrelevant ones. In previous studies, various rank loss functions have been developed for ranking-based learning (Lapin, Hein, & Schiele, 2017). To demonstrate the effectiveness of our joint latent ranking embedding framework, we adopt two simple yet typical rank loss functions, *RankNet loss* (Burges et al., 2005) and the margin-based *hinge rank loss* (Herbrich, Obermayer, & Graepel, 1999), in our work although other rank loss functions (Lapin et al., 2017) may be employed in our framework as well. RankNet loss (Burges et al., 2005) provides a generic loss function for ranking-based learning from a probabilistic perspective, while hinge rank loss (Herbrich et al., 1999) was originally proposed for structural SVMs and has been widely used in different tasks including single-label zero-shot learning (e.g., Akata et al., 2015; Frome et al., 2013).

Nevertheless, we observe the following phenomenon in our experiments when using the original RankNet and hinge rank losses. By using only a ranking constraint in either of two rank losses, all the labels are considered independently and treated equally so that the less frequently used relevant labels might be overlooked during learning. Moreover, two rank losses generally make use of pairwise constraints to explore a relationship between labels associated with an instance explicitly. However, the relatedness scores in such rank losses are not bounded and could hence vary across different examples. Thus, some "difficult" pairs of labels are likely to incur a larger cost that predominates the overall loss, which could make the learning biased to those pairs of labels only. Furthermore, relatedness scores may vary in a large range for different training examples even though proper ranking relationships among them are established, which results in the poor performance. Motivated by the above observation, we introduce a regularization term to RankNet and hinge rank losses to overcome those problems.

For the target label set expressed with binary indicators, $\boldsymbol{y}_i$, in the ith training example, $(\boldsymbol{x}_i, \boldsymbol{y}_i)$, the elements of +1 indicate all the labels relevant to $\boldsymbol{x}_i$ while elements of -1 express all the remaining labels irrelevant to $\boldsymbol{x}_i$ in terms of all the known actions in C^{Tr}. Likewise, $\boldsymbol{y}^c$, a binary indicator in $\{+1, -1\}$ regarding whether the cth action appears in training examples in $\mathcal{D}$, can be handled in the same manner. Thus, the relatedness scores of $\boldsymbol{x}_i$ to its positive and negative labels, $\boldsymbol{o}_i$, are achieved with Eqs. (2) and (3), and the relatedness scores of the cth label to all the training examples, $\boldsymbol{o}^c$, are calculated with Eqs. (1) and (5). Based on the above quantities, we can define our regularized rank loss functions, $l_v(\boldsymbol{o}_i, \boldsymbol{y}_i)$ and $l_s(\boldsymbol{o}^c, \boldsymbol{y}^c)$.

Formally, we define the regularized RankNet loss function for visual embedding of $\boldsymbol{x}_i$ as follows:

$$l_v(\boldsymbol{o}_i, \boldsymbol{y}_i) = \omega_i \Big(\sum_{p \in C_i^{Tr+}} \sum_{q \in C_i^{Tr-}} \log\big(1 + \exp(o_{iq} - o_{ip})\big) + \sum_{j \in C^{Tr}} \log\big(1 + \exp(-y_{ij} o_{ij})\big) \Big), \tag{8}$$

where $\omega_i = (|C_i^{Tr+}| \cdot |C_i^{Tr-}| + |C^{Tr}|)^{-1}$ normalizes this per-instance regularized rank loss. Corresponding to the elements of +1 and -1 in $\boldsymbol{y}_i$, C_i^{Tr+} and C_i^{Tr-} denote two subsets of relevant and irrelevant labels to $\boldsymbol{x}_i$, respectively. Intuitively, minimizing the first term in Eq. (8) ensures that all the labels relevant to $\boldsymbol{x}_i$ are ranked ahead of those irrelevant to $\boldsymbol{x}_i$. The second term in Eq. (8) plays a regularization role; minimizing this term during learning promotes the relatedness scores by enlarging the relatedness scores to the relevant labels as well as diminishing those to the irrelevant ones simultaneously, which tackles the problems observed in our experiments.

Likewise, we define the regularized RankNet loss function for semantic embedding of label c as follows:

$$l_s(\boldsymbol{o}^c, \boldsymbol{y}^c) = \omega_c \Big(\sum_{p \in \mathcal{D}^{c+}} \sum_{q \in \mathcal{D}^{c-}} \log\big(1 + \exp(o_q^c - o_p^c)\big) + \sum_{j \in \mathcal{D}} \big(1 + \exp(-y_j^c o_j^c)\big) \Big), \tag{9}$$

where $\omega_c = (|\mathcal{D}^{c+}| \cdot |\mathcal{D}^{c-}| + |\mathcal{D}|)^{-1}$ normalizes the per-class regularized rank loss. $\mathcal{D}^{c+}$ and $\mathcal{D}^{c-}$ are the positive and the negative training example subsets, respectively, regarding the cth label. With the same treatment as used in Eq. (8), minimizing Eq. (9) ensures that the video instances conveying the action of the cth label are ranked above all those without this action. Moreover, those video instances conveying the action of the cth label, indicated by $y_j^c = +1$, tend to have as high relatedness scores as possible while all other video instances without this action, indicated by $y_j^c = -1$, tend to have as low relatedness scores as possible.

Similarly, we define a regularized hinge rank loss function for visual embedding of $\boldsymbol{x}_i$ as follows:

$$l_v(\boldsymbol{o}_i, \boldsymbol{y}_i) = \omega_i \Big(\sum_{p \in C_i^{Tr+}} \sum_{q \in C_i^{Tr-}} \max\big(0, m - o_{ip} + o_{iq}\big) + \sum_{j \in C^{Tr}} \max\big(0, m - y_{ij} o_{ij}\big) \Big), \tag{10}$$

where $\omega_i = (|C_i^{Tr+}| \cdot |C_i^{Tr-}| + |C^{Tr}|)^{-1}$ and m is a pre-specified margin. Thus, minimizing the first term in Eq. (10) ensures that all the labels relevant to $\boldsymbol{x}_i$ are ranked ahead of those irrelevant to $\boldsymbol{x}_i$ with a pre-specified margin, m. The second term in Eq. (10) plays a regularization role; minimizing this term during learning promotes the margin-based relatedness scores by enlarging the relatedness scores to the relevant labels as well as diminishing those to the irrelevant ones simultaneously.

Likewise, we define a regularized hinge rank loss function for semantic embedding of label c as follows:

$$l_s(\boldsymbol{o}^c, \boldsymbol{y}^c) = \omega_c \Big(\sum_{p \in \mathcal{D}^{c+}} \sum_{q \in \mathcal{D}^{c-}} \max\big(0, m - o_p^c + o_q^c\big) + \sum_{j \in \mathcal{D}} \max\big(0, m - y_j^c o_j^c\big) \Big), \tag{11}$$

where $\omega_c = (|\mathcal{D}^{c+}| \cdot |\mathcal{D}^{c-}| + |\mathcal{D}|)^{-1}$ and m is a pre-specified margin. With the same treatment as used in Eq. (10), minimizing Eq. (11) ensures that the video instances conveying the action of the cth label are ranked above all those without this action with a margin, m. Moreover, those video instances conveying the action of the cth label, indicated by $y_j^c = +1$, tend to have as high relatedness scores as possible while all other video instances without this action, indicated by $y_j^c = -1$, tend to have as low relatedness scores as possible.

As a result, we can employ either our regularized RankNet loss functions in Eqs. (8) and (9) or the regularized hinge rank loss functions in Eqs. (10) and (11) to train visual and semantic embedding models in our framework.

3.2.3. Learning algorithm

As formulated in Eqs. (6) and (7), learning is going to find the optimal parameters, Θ_v^* and Θ_s^*, by minimizing two loss functions, $l_v(\boldsymbol{o}_i, \boldsymbol{y}_i)$ and $l_s(\boldsymbol{o}^c, \boldsymbol{y}^c)$, defined in Section 3.2.2. However, the relatedness scores required in $l_v(\boldsymbol{o}_i, \boldsymbol{y}_i)$ regarding the visual model involve the output of the semantic model, E^s, and vice versa (c.f. Fig. 1). Moreover, $l_v(\boldsymbol{o}_i, \boldsymbol{y}_i)$ requires the relatedness scores between all the candidate labels and each of training examples, while $l_s(\boldsymbol{o}^c, \boldsymbol{y}^c)$ needs the relatedness scores between all the training examples and each of all the action labels in C^{Tr}. Thus, our optimization problems are very complex and unsolvable simultaneously with commonly used local search methods, e.g., gradient-descent based methods.

Motivated by the works dealing with similar optimization problems (e.g., Jiang, Wu, Wang, Xue, & Chang, 2017; Kavukcuoglu, Ranzato, & LeCun, 2010), we come up with a learning algorithm to train the visual and the semantic models alternately during learning. In our alternate learning strategy, our learning algorithm begins with randomly initializing the parameters in the semantic model and then use the initialized parameter to generate the initial semantic embedding. By using the initial semantic embedding in $l_v(\boldsymbol{o}_i, \boldsymbol{y}_i)$, the visual model can be trained with a local search method such as the mini-batch stochastic gradient decent method. After one epoch, the current parameters in the visual model are frozen and used to generate the visual embedding for all the examples. By using the current visual embedding in $l_s(\boldsymbol{o}^c, \boldsymbol{y}^c)$, the semantic model is trained in the same manner. This alternate learning process carries on until a stopping condition is satisfied. The details of this *alternate* learning algorithm are described in Algorithm 1.

Algorithm 1 Joint Latent Ranking Embedding Learning

Input: Randomly initialize parameters, Θ_v^0 and Θ_s^0, in the visual and the semantic models, respectively; extract the visual representations of training example, $\boldsymbol{x}_i$, $i = 1, \cdots, N$, and the semantic representations of all the training labels, $\boldsymbol{s}_c$, $\forall c \in C^{Tr}$; input the target label matrix of the training set, Y; pre-set the dimensionality of joint latent ranking embedding space, d_e.

Output: Optimal model parameters: Θ_v^* and Θ_s^*.

1: Generate the initial semantic embedding $\boldsymbol{\phi}_s(\boldsymbol{s}_c; \Theta_s^0)$, $\forall c \in C^{Tr}$; $t \leftarrow 0$.
2: **repeat**
3: $t \leftarrow t + 1$;
4: $\Theta_v^t = \text{argmin}_{\Theta_v} \sum_{i=1}^N l_v(\boldsymbol{o}_i, \boldsymbol{y}_i)$ with the current semantic embedding for one epoch;
5: Generate the visual embedding with the current visual model, $\boldsymbol{\phi}_v(\boldsymbol{x}_i; \Theta_v^t)$, $i = 1, \cdots, N$;
6: $\Theta_s^t = \text{argmin}_{\Theta_s} \sum_{c \in C^{Tr}} l_s(\boldsymbol{o}^c, \boldsymbol{y}^c)$ with the current visual embedding for one epoch;
7: Generate the semantic embedding with the current semantic model $\boldsymbol{\phi}_s(\boldsymbol{s}_c; \Theta_s^t)$, $\forall c \in C^{Tr}$;
8: **until** Stopping condition is met.
9: $\Theta_v^* \leftarrow \Theta_v^t$ and $\Theta_s^* \leftarrow \Theta_s^t$.

It is worth stating that two rank loss functions defined for visual and semantic model are related and the optimization of one model would naturally promote the other towards its optimal solution. Hence, our alternate learning algorithm may converge after running finite epochs with the same properties held for similar methods (Jiang et al., 2017; Kavukcuoglu et al., 2010).

3.3. Multi-label zero-shot recognition

Once the joint latent ranking embedding learning is completed, we obtain a mapping function: $\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{c}) = \{\boldsymbol{\phi}_v(\boldsymbol{x}|\Theta_v^*), \boldsymbol{\phi}_s(\boldsymbol{c}|\Theta_s^*)\}$ where $\boldsymbol{\phi}_v(\boldsymbol{x}|\Theta_v^*)$ and $\boldsymbol{\phi}_s(\boldsymbol{c}|\Theta_s^*)$ are the visual and the semantic embedding functions implemented by the trained visual and semantic models, respectively. Then, we can use this mapping function for multi-label zero-shot human action recognition.

For recognition, we first extract the semantic representations of all the labels, including both *known* and *unseen* labels, in a considered label collection: $\boldsymbol{s}_c$, $\forall c \in C$; $C = C^{Tr} \cup C^U$ and $C^{Tr} \cap C^U = \emptyset$. By using the semantic embedding function, we achieve the semantic embedding of all the labels: $\hat{\boldsymbol{e}}_c^s = \boldsymbol{\phi}_s(\boldsymbol{s}_c|\Theta_s^*)$, $\forall c \in C$. For a test video clip, we divide it into T segments and extract its segment-level representations collectively denoted by $\hat{\boldsymbol{x}} = \{\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2, \ldots, \hat{\boldsymbol{x}}_T\}$. Technical details for extracting semantic and visual representations can be found in Section 4.2. By feeding $\hat{\boldsymbol{x}}$ to the visual embedding function, we achieve its visual embedding: $\hat{\boldsymbol{e}}^v = \boldsymbol{\phi}_v(\hat{\boldsymbol{x}}|\Theta_v^*)$. Thus, the relatedness scores between this test video clip, $\hat{\boldsymbol{x}}$, and all the actions in the considered label collection C, including known and unseen labels during learning, is achieved by

$$S(\hat{\boldsymbol{x}}, c) = \langle \hat{\boldsymbol{e}}^v, \hat{\boldsymbol{e}}_c^s \rangle, \quad \forall c \in C. \tag{12}$$

Finally, we achieve a ranking action label list, $L(\hat{\boldsymbol{x}})$, for this test video clip by sorting its relatedness scores measured against all

the labels in C with Eq. (12):

$$L(\hat{\boldsymbol{x}}) = \left\{c_i\right\}_{i=1}^{|C|}, \tag{13}$$

where $\forall c_i, c_j \in C$, $\text{Score}(\hat{\boldsymbol{x}}, c_i) \geq \text{Score}(\hat{\boldsymbol{x}}, c_j)$ if $i < j$.

In our experiments regarding the use of two different rank losses in our framework, we observe that the regularized RankNet and hinge rank losses often behave differently in several evaluation scenarios (c.f. Section 4). Although two rank losses generally yield the comparable performance overall, a closer look suggests that those correctly recognized video instances are quite different when two different rank losses are used in our framework, respectively. Hence, we would employ a simple fusion method to exploit the complementary aspect resulted from the use of two different rank losses. In order to fuse the results yielded by the models trained with two different rank losses, we first normalize the relatedness scores of $\hat{\boldsymbol{x}}$ to the considered label collection, C, generated by each of two models as follows:

$$\tilde{\text{S}}(\hat{\boldsymbol{x}}, c) = \frac{\text{S}(\hat{\boldsymbol{x}}, c) - \text{S}_{min}}{\text{S}_{max} - \text{S}_{min}}, \tag{14}$$

where S_{max} and S_{min} are the highest and lowest related scores of video clips, respectively, measured on a test set. Let $\tilde{\text{S}}^{(Reg)}(\hat{\boldsymbol{x}}, c)$ and $\tilde{\text{S}}^{(Hinge)}(\hat{\boldsymbol{x}}, c)$ denote the normalized relatedness scores yielded by two models trained with our regularized rank and the hinge rank losses, respectively. Then, the fused relatedness scores, $\tilde{\text{S}}^{(Fusion)}(\hat{\boldsymbol{x}}, c)$ is simply an average between $\tilde{\text{S}}^{(RankNet)}(\hat{\boldsymbol{x}}, c)$ and $\tilde{\text{S}}^{(Hinge)}(\hat{\boldsymbol{x}}, c)$; i.e.,

$$\tilde{\text{S}}^{(Fusion)}(\hat{\boldsymbol{x}}, c) = \frac{\tilde{\text{S}}^{RankNet}(\hat{\boldsymbol{x}}, c) + \tilde{\text{S}}^{Hinge}(\hat{\boldsymbol{x}}, c)}{2}. \tag{15}$$

Based on the fused relatedness scores, a ranking action label list, $L^{(Fusion)}(\hat{\boldsymbol{x}})$, is achieved in the same manner as specified in Eq. (13) for any test video clip, $\hat{\boldsymbol{x}}$.

4. Experimental setting

In this section, we describe our experimental design and settings, including datasets, visual and semantic representations, model learning, evaluation scenarios and criteria used in our experiments. Moreover, we design a number of comparative experiments to exhibit the gain resulting from different components in our framework and to demonstrate the effectiveness of our framework by a comparison to several state-of-the-art multi-label ZSL methods that could be applied to general human action recognition.

4.1. Datasets and splits

We first describe datasets and their split settings used in our experiments for simulation of multi-label ZSL scenarios.

4.1.1. Datasets

To evaluate our framework, we employ two publicly available video datasets: *Breakfast* (Kuehne, Arslan, & Serre, 2014) and *Charades* (Sigurdsson et al., 2016), in our experiments. In both datasets, at least two actions are involved in each video clip and the duration of each video clip is relatively long, which implies the temporal coherence information may be explored and exploited in human action recognition. Hence, both datasets are suitable to evaluate weakly annotated multi-label human action recognition. Below, we summarize the main aspects of two video datasets.

Breakfast: In this dataset (Kuehne et al., 2014), there are 1989 video clips totally, where a video clip conveys several cooking actions. Totally, there are 49 cooking actions (excluding the "silence" label), such as 'stirring", "pouring_milk" and "opening_the_fridge". Those actions are performed by 52 people in different kitchens. Although this dataset is not collected especially for multi-label human action recognition, we would use it as a proof-of-concept test bed.

Charades: This dataset (Sigurdsson et al., 2016) is collected from hundreds of people recording videos in their own home especially for video-based human activity analysis in daily lives. Hence, it is very challenging for multi-label human action recognition. In this dataset, there are 9848 video clips involving 157 different human actions totally, acting out casual everyday activities. An average duration of video clips is around 30 s and an average number of actions involved in a video clip is 6.8. Those actions are performed by 267 people from three continents, and more than one person appear in over 15% of all the video clips. The raw video data (scaled to 480p) are used in our experiments, which are available from the Charades project page.[2]

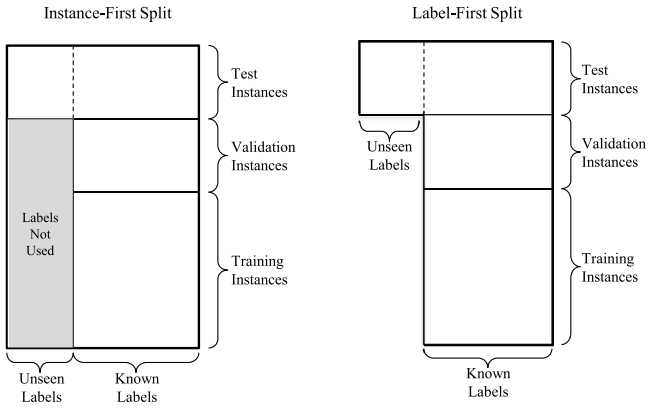

Fig. 2. Two different data split settings used in our experiments. In the *instance-first* split (left plot), a dataset is simply split into three mutually exclusive subsets: training, validation and test subsets. Any unseen labels associated with instances in training and validation subsets (shaded area) are removed from their target label sets before being used in learning; i.e., only the subset of known target labels in a training example are used in the learning. In the *label-first* split (right plot), a number of labels are first specified as unseen labels. Instances associated with any of unseen labels form a test subset. The remaining data of known labels are further divided into training and validation subsets to be used in learning.

4.1.2. Dataset splits

To simulate a zero-shot scenario, we need to split a dataset into training and test sets where a training set contains examples associated with only known classes while a test set has test instances involving at least one unseen class. Unlike single-label ZSL where a dataset is automatically split into training and test sets once unseen classes are specified, the dataset split issue in multi-label ZSL becomes much more complicated. In our experiments, we make two different split settings, *instance-first split* (IFS) and *label-first split* (LFS), as illustrated in Fig. 2.

Instance-First Split

This is a commonly used data split setting in all the existing multi-label ZSL works (e.g., Mensink et al., 2014; Nam et al., 2015; Zhang, Gong, and Shah, 2016). In this setting, instances in a dataset are first split into training, validation and test subsets. The training and the validation subsets are used for parameter estimation and hyper-parameter tuning, the dimension of latent embedding space d_e and the number of iterations of Algorithm 1 for our model. The test set that may or may not involve unseen labels is reserved for performance evaluation. Then, we divide the

[2] http://allenai.org/plato/charades/.

Table 2
Information on two different data split settings.

Dataset	Split method	# Training inst.	# Validation inst.	# Test inst.	# Known labels	# Unseen labels
Breakfast	Instance-first	1196	126	667	39	10
	Label-first	1019/917/823	200	770/872/966	39	10
Charades	Instance-first	6385	1600	1863	117	40
	Label-first	4580/4176/3987	1000	4268/4672/4861	137	20

action label collection into mutually exclusive known and unseen label sets. Before learning, any unseen labels in the target label set of an instance in the training and the validation subsets are removed as shown in the left plot of Fig. 2. In other words, only known labels in the target label sets of an instance in those two datasets are used in learning. It is worth clarifying that unlike single-label ZSL, it is often infeasible to simulate a ZSL scenario by manipulating the validation set due to insufficient data in two datasets used in our experiments. Hence, the validation for hyper-parameter tuning in our experiments has to follow the typical protocol used in multi-label learning (Zhang & Zhou, 2014).

To split the Breakfast dataset with this setting, we adopt the pre-split by data collectors (Kuehne et al., 2014), where the video clips of 13 people are reserved for test. We further divide the rest video clips for training and validation: video clips of 32 people for training and the remaining video clips of seven people for validation. As a result, the numbers of video clips for training, validation and test are 1196, 126 and 667, respectively. Then we randomly split the 49 labels into known and unseen labels: 10 labels reserved as unseen labels and the rest 39 as known labels.

For the Charades dataset, we also adopt its pre-split provided by data collectors (Sigurdsson et al., 2016), where 7985 and 1863 video clips are used for training and test, respectively. We further divide training data into two subsets: 6385 for training and 1600 for validation in our experiments. Then we randomly choose 40 out of 157 human actions as unseen classes and the rest 117 human actions are hence known actions.

Label-First Split

Although the instance-first data split setting is widely used in multi-label ZSL, it suffers from a fundamental limitation. It is well known that multiple labels together could frame a specific concept and removing any label from this label cohort may lead to a less accurate semantic meaning and biases in learning. Furthermore, the instance-first split allows for accessing to visual features of instances involved in unseen actions. To overcome this limitation, we propose a novel data split setting for multi-label ZSL named *label-first* split. In this new setting, all the labels in a label collection used in a dataset is first divided into two mutually exclusive subsets: known and unseen labels. Then, all the instances having any unseen labels are reserved for test and the rest instances of known labels only are further divided into two subsets for training and validation, as shown in the right plot of Fig. 2. Due to sparsity of training data, the validation in the label-first split also adopts the protocol used in multi-label learning (Zhang & Zhou, 2014).

To split the Breakfast dataset with this setting, we randomly choose 10 labels for unseen labels and the rest 39 labels are designated as known labels accordingly. Hence, this dataset is naturally split into two sets for training and test. The training data are further divided for training and validation. For validation, we randomly choose 200 instances from the training data. Likewise, the Charades dataset is split by using 20 randomly chosen label for unseen labels. Thus, the remaining 137 labels become known labels. From the instances of known labels, we randomly choose 1000 instance used for validation. It is worth stating that the current datasets do not allow for reserving a large number of classes as unseen classes in either the IFS or the LFS setting. In the IFS, the more labels reserved as unseen labels, the less accurate mapping learned from visual to semantic domains due to the existence of visual features of unseen actions and a lack of the corresponding action labels in such training examples. In the LFS, the more labels reserved as unseen labels, the fewer training examples available. Hence, the training examples do not convey the essential information required in learning.

For reliability, we repeat our experiments on each dataset under each split setting for three trials. During a trial, training data given in the pre-split of each dataset is randomly divided into training and validation subsets with the instance-first split setting, and a known/unseen label split on each dataset is chosen randomly with the label-first split setting. For clarity, we summarize the data split information[3] on two datasets in Table 2.

4.2. Visual and semantic representations

In our experiments, we use visual representations extracted with the existing C3D deep network (Tran, Bourdev, Fergus, Torresani, & Paluri, 2015) and word vectors as semantic representations (Mikolov et al., 2013).

As suggested by Tran et al. (2015), the C3D features are extracted for a segment of 16 frames with eight frames overlapping between two adjacent segments. Thus, a training/test video clip is always divided into T segments with the treatment as follows. To ensure that each video clip can be divided into T segments, any video clip must have $8 * (T + 1)$ frames. To this end, we simply down-sample those video clips of more frames with a proper sampling rate so that T C3D feature vectors can be extracted and collectively form a *segment-based* visual representation for this video clip. When a video clip has fewer frames, we first extract C3D feature vectors from those frames in this video clip and then pad all-zero vectors to the visual representation until there are T feature vectors. Also, we can convert T feature vectors into a holistic *instance-level* visual representation via averaging those T feature vectors. By using such an instance-level visual representation in our comparative study, we would demonstrate a performance gain benefiting from exploring/exploiting temporal coherence information underlying segments in a video clip. In our experiments, the segment-based visual representation is always used in our model while the instance-level visual representation is used in the baseline and the state-or-the-art models (c.f. Section 4.6) unless a different setting is specified. Based on a cross-validation experiment, we choose $T = 300$ for Breakfast and $T = 20$ for Charades. Although only C3D features are used in our experiments, it is worth mentioning that other kinds of visual representations, e.g., IDT features (Wang & Schmid, 2013) and deep image features extracted on a frame basis, can also be used straightforwardly in our framework.

In our experiments, we adopt Word2Vec as our semantic representation. Word2Vec was trained with a skip-gram neural network on the Google News dataset of 100 billion words (Mikolov et al., 2013). As a result, one action label is represented by a 300-dimensional word vector. Although only 300-dimensional word vectors are used in our experiments, word vectors of different

[3] All the data splits and source code used in our experiments are available on our project website: http://staff.cs.manchester.ac.uk/~kechen/MLZSHAR.

dimensionality may be used, and moreover, other semantic representations, e.g., attributes, can be used in our framework without any difficulty if available.

It is worth emphasizing that the same treatment described above is applied in both the learning and the recognition phases to extract visual and semantic representations.

4.3. Model learning

In our experiments, model learning is implemented on Keras (Chollet, 2015), a high-level neural networks library, running on top of either TensorFlow or Theano. As we use two neural networks to carry out the visual and the semantic models (c.f. the left box in Fig. 1), we need to decide the specific network architectures and relevant hyper-parameters on two datasets during the model learning. The optimal hyper-parameters are found by a grid-based search via a cross-validation procedure. The Adam (Kingma & Ba, 2014), a stochastic optimization method, is used for training our model with its default configuration.

The visual model takes a sequence of segment-level C3D representations of $d_x = 4096$ features as input to the LSTM layer where there are N_1 LSTM units. To improve the generalization, we also apply the *dropout* procedure (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014) to the LSTM layer where a dropout rate needs specifying. The output of LSTM units are fed to a fully connected dense layer of N_2 neurons, and the output of this dense layer are further fed to the visual embedding layer of d_e neurons. During learning, there are no hyper-parameters involved in the score and the average pooling layer in the visual model.

As described in Section 3.1, the semantic model is carried out by a fully-connected three-layer feed-forward neural network. The word vectors of $d_s = 300$ dimensions are first input to a hidden layer of N_1 *ReLu* units. Subsequently, the output of this hidden layer are fed to the semantic embedding layer of d_e linear units. Note that for joint latent ranking embedding learning, the dimension of the semantic embedding space is set to the same of the visual embedding space in our experiments. Likewise, there are no hyper-parameters involved in the score and the average pooling layer in the semantic model during learning.

4.4. Evaluation scenarios

Multi-label zero-shot recognition is complex given the fact that a test instance may be associated with a label set including both known and unseen class labels. Thus, there are different evaluation scenarios in previous works (Sandouk & Chen, 2016b; Zhang, Gong, and Shah, 2016); each focuses on a specific aspect. Following their settings (Sandouk & Chen, 2016b; Zhang, Gong, and Shah, 2016), we evaluate our framework along with other learning models used in our comparative study described later on in this section in three different scenarios:

Known-action only: In this setting, the performance is evaluated regarding only known (training) actions. This scenario boils down to the conventional supervised multi-label learning. In this circumstance, we no longer take any unseen action labels into account during test; for a test instance, its relatedness score ranking list contains only those regarding known action labels in C^{Tr} and any unseen action label in C^U in its ground-truth label set, if there is, will be removed such that the modified ground-truth set includes only known action labels in C^{Tr}.

Unseen-action only: In this setting, the performance is evaluated regarding only unseen (test) actions. This scenario boils down to a standard ZSL setting. In this situation, we no longer consider any known action labels; for a test instance, its relatedness score ranking list contains only those regarding unseen action labels in C^U and any known action label in C^{Tr} in its ground-truth label set, if there is, will be removed such that the modified ground-truth set includes only unseen action labels in C^U.

Generalized ZSL: In this setting, the performance is evaluated regarding all the actions of which labels appearing in a label collection C without considering if an action label is known or unseen during learning. This scenario has been named *generalized ZSL* in the machine learning community. In this situation, both known and unseen action labels are treated equally; for a test instance, its relatedness score ranking list contains those regarding all the action labels in C and the evaluation is made against its ground-truth label set that could be a mixture of known and unseen labels. It is worth highlighting that the generalized ZSL setting is required by multi-label zero-shot human action recognition in a real application.

4.5. Evaluation metrics

There are a variety of evaluation metrics for multi-label learning. Depending on the output of a multi-label learning system, the evaluation metrics are generally divided into two types: *ranking-based* and *bipartition-based* metrics (Nam et al., 2015; Sorower, 2010). Ranking-based metrics work for the situation that a learning system yields a ranking list of continuous-valued relatedness scores on all the candidate labels. In contrast, bipartition-based metrics are used when a learning system produces only a binary indicator vector for all the candidate labels, where 1/0 element expresses the presence/absence. Since our model yields a ranking list of continuous-valued relatedness scores, we employ two commonly used ranking-based metrics for performance evaluation (Li et al., 2016; Mensink et al., 2014; Nam et al., 2015; Sorower, 2010; Zhang, Gong, and Shah, 2016), *I*nstance-centric *M*ean *A*verage *P*recision (I-MAP) and *L*abel-centric *M*ean *A*verage *P*recision (L-MAP). In addition, we employ other metrics, *precision*, *recall* and F_1 score, which have also been used in the performance evaluation of multi-label learning (Gong, Jia, Leung, Toshev, & Ioffe, 2013; Zhang & Zhou, 2014).

To facilitate our presentation, we first define the *precision-at-k* (Manning, Raghavan, & Schütze, 2009) in a generic form:

$$P@k(A, B) = \frac{1}{k}\left|A \cap B[1, \ldots, k]\right|, \tag{16}$$

where A is a ground-truth set, B is a set of all the retrieved entities ranked in terms of relevance, and $B[1, \ldots, k]$ indicates top k entities in B. Given a test dataset, $\mathbb{D}_T = \{\hat{\boldsymbol{x}}_i\}_{i=1}^{|\mathbb{D}_T|}$, a learning model yields a *label-based* ranking list for a test instance, $\hat{\boldsymbol{x}}_i \in \mathbb{D}_T$, in terms of its relatedness scores to all the labels in C (c.f. Eqs. (12) and (13)): $L(\hat{\boldsymbol{x}}_i) = \{c_j\}_{j=1}^{|C|}$, where $\forall c_p, c_q \in C$, $\text{Score}(\hat{\boldsymbol{x}}_i, c_p) \geq \text{Score}(\hat{\boldsymbol{x}}_i, c_q)$ if $p < q$. Let $C(\hat{\boldsymbol{x}}_i)$ denote the ground-truth label set of $\hat{\boldsymbol{x}}_i$. I-MAP over a test dataset $\mathbb{D}_T$ is defined by

$$\text{I–MAP} = \frac{1}{|\mathbb{D}_T|}\sum_{i=1}^{|\mathbb{D}_T|}\frac{\sum_{c=1}^{|C|} P@c\Big(C(\hat{\boldsymbol{x}}_i), L(\hat{\boldsymbol{x}}_i)\Big)\delta\big(c, C(\hat{\boldsymbol{x}}_i)\big)}{|C(\hat{\boldsymbol{x}}_i)|}, \tag{17}$$

where $\delta\big(c, C(\hat{\boldsymbol{x}}_i)\big) = 1$ if $c \in C(\hat{\boldsymbol{x}}_i)$ and $\delta\big(c, C(\hat{\boldsymbol{x}}_i)\big) = 0$ otherwise.

While I-MAP measures the accuracy in terms of test instances, L-MAP is used to evaluate the performance from a different perspective in light of candidate labels. Given a specific label $c \in C$, a model predicts the relatedness scores against the action specified by the cth label for all the test instances in $\mathbb{D}_T$. Hence, we can achieve an *instance-based* ranking list for the cth label, $L^c = \{\hat{\boldsymbol{x}}_{i_j}\}_{j=1}^{|\mathbb{D}_T|}$, in terms of their relatedness scores against the cth label where $\forall \hat{\boldsymbol{x}}_{i_p}, \hat{\boldsymbol{x}}_{i_q} \in \mathbb{D}_T$, $\text{Score}(\hat{\boldsymbol{x}}_{i_p}, c) \geq \text{Score}(\hat{\boldsymbol{x}}_{i_q}, c)$ if $p < q$. Let D^c denote the collection of those test instances of which their

ground-truth label sets indeed include the cth label. Thus, the L-MAP over a test dataset, $\mathbb{D}_T$, is defined by

$$\text{L–MAP} = \frac{1}{|C|}\sum_{c=1}^{|C|}\frac{\sum_{i=1}^{|\mathbb{D}_T|} P@i\Big(D^c, L^c\Big)\delta\big(\hat{\boldsymbol{x}}_i, D^c\big)}{|D^c|}, \tag{18}$$

where $\delta(\hat{\boldsymbol{x}}_i, D^c) = 1$ if $\hat{\boldsymbol{x}}_i \in D^c$ and $\delta(\hat{\boldsymbol{x}}_i, D^c) = 0$ otherwise.

Those widely used evaluation metrics in information retrieval have also been used in evaluating multi-label learning systems (e.g., Gong et al., 2013; Zhang & Zhou, 2014). In our experiments, we adopt overall top-k precision, recall and F_1 score measured over a test dataset, $\mathbb{D}_T$, which are defined as follows:

$$\text{precision}(k) = \frac{\sum_{i=1}^{|\mathbb{D}_T|} P@k\big(C(\hat{\boldsymbol{x}}_i), L(\hat{\boldsymbol{x}}_i)\big)}{k * |\mathbb{D}_T|}, \tag{19}$$

$$\text{recall}(k) = \frac{\sum_{i=1}^{|\mathbb{D}_T|} P@k\big(C(\hat{\boldsymbol{x}}_i), L(\hat{\boldsymbol{x}}_i)\big)}{\sum_{i=1}^{|\mathbb{D}_T|} |C(\hat{\boldsymbol{x}}_i)|}, \tag{20}$$

$$\text{F}_1(k) = \frac{2 * \text{precision}(k) * \text{recall}(k)}{\text{precision}(k) + \text{recall}(k)}. \tag{21}$$

4.6. Comparative study

In our experiments, we systematically conduct a comparative study from two different perspectives: ablation study and state-of-the-art models. As a result, a number of baseline systems are designed to demonstrate roles played by the main components in our framework while several state-of-the-art multi-label ZSL algorithms are adapted for human action recognition. For the comparative study, we evaluate each of different models on three evaluation scenarios with evaluation metrics described in Sections 4.4 and 4.5 under the exactly same conditions, including visual and semantic representations. As there are alternative pooling strategies that could be used to implement our framework, we further investigate those pooling strategies by comparing them to the average pooling used in our framework.

4.6.1. Baseline models

To investigate the roles played by different component mechanisms employed in our framework, we design four baseline models, *random guess of scores*, *non-recurrent connection*, *without semantic embedding* and *randomized label representation*, by manipulating our framework with different purposes described as follows:

Random guess of scores (RGS): This is a general baseline that provides a lowest performance bound used for a reference to improvement made by a learning model. In our work, we randomly generate relatedness scores of all the candidate labels for a test instance. Then the performance of this baseline model is evaluated based on the random guess of scores. For reliability, we repeat the RGS process 100 times in our experiments and the statistics of the RGS performance including mean and standard error of mean (SEM) are reported.

Non-recurrent connection (NRC): In our framework, a LSTM layer of recurrent connections is employed to capture temporal coherence underlying sequential video data in the visual embedding learning. To examine the role played by the LSTM layer, we replace the recurrent connected layer with a fully connected layer without recurrent connections and keep all other components in our framework unchanged. By this setting, our model is converted into a baseline model named *non-recurrent connection*. During learning, obviously, this baseline model no longer explicitly makes use of the temporal dependency information underlying sequential segments in a video clip. Algorithm 1 is used directly for parameter estimation.

Without semantic embedding (WSE): In our framework, there is a semantic model for semantic embedding with the motivation that the use of a joint latent ranking embedding space narrows the semantic gap between visual and semantic domains and the zero-shot recognition should be done in the joint latent ranking embedding space. However, some existing works, e.g., Fast0Tag (Zhang, Gong, and Shah, 2016), do not learn a semantic embedding and the zero-shot recognition takes place directly in the semantic space. To examine the effectiveness of our semantic embedding, we come up with a baseline model named *without semantic embedding* by removing the semantic model from our framework. Thus, the original semantic representations are used to replace the semantic embedding representations, $\boldsymbol{E}^s$, required by the score layer in the visual model, which amounts to mapping the visual space directly onto the original semantic space. As this baseline model has only the visual model, the learning becomes simpler; i.e., solving the optimization problem formulated in Eq. (6) based on the original semantic representation with the Adam (Kingma & Ba, 2014). It is worth clarifying that this baseline model is similar to Fast0Tag (Zhang, Gong, and Shah, 2016) apart from an LSTM-based visual embedding model and the segment-level visual representation used in this baseline model while a feed-forward neural network and instance-level visual representation are employed by Fast0Tag for visual embedding.

Randomized label representation (RLR): One of the most important issues in ZSL is exploring/exploiting the side information conveyed in the semantic domain. As our framework works for multi-label zero-shot recognition, we would investigate whether the semantic relatedness information encoded in the semantic embedding, inherited from the original semantic representations, is effectively used in knowledge transfer. To this end, we design another baseline model named *randomized label representation* by replacing the word vector of a label with a vector of the same dimensionality that is generated randomly and normalized with the l_2 norm to ensure that it has the same range as that of the word vector. Apparently, the semantic relatedness information no longer exists in such randomized label representations. For parameter estimation, Algorithm 1 is used directly via replacing the semantic representations of labels with the randomized label representations in training data.

4.6.2. State-of-the-art methods

Although, to the best of our knowledge, there exists no work in multi-label zero-shot human action recognition, we notice that there are a few multi-label ZSL algorithms. In our comparative study, we adopt and extend those multi-label ZSL algorithms for human action recognition for a thorough evaluation of our proposed framework. Below, we briefly describe those multi-label ZSL algorithms used in our experiments.

Direct Semantic Prediction (DSP): DSP is a well-known baseline model used in previous works for multi-label ZSL (e.g., Sandouk & Chen, 2016b). DSP is derived from *direct attribute prediction* originally proposed for single-label ZSL (Lampert et al., 2014). The idea behind DSP is learning a mapping function $\phi : X \rightarrow \mathcal{S}$ from visual to semantic space directly for ZSL. In our experiments, we employ support vector regressor models to learn the mapping function $\phi(\cdot)$.

Convex combination of Semantic Embedding (ConSE): ConSE is a ZSL algorithm proposed by Norouzi et al. (2014), which can be naturally applied to multi-label ZSL. As same as formulated in DSP, ConSE also learns a mapping to predict a compositional semantic representation from the visual representation of a given video clip.

COSTA: COSTA is a method proposed by Mensink et al. (2014) especially for multi-label zero-shot classification. In this method,

multi-label classification is converted into a number of binary classification problems via a one-vs-rest setting. $|C^{Tr}|$ linear binary SVMs are trained based on the examples regarding $|C^{Tr}|$ known actions. Then the parameters of the SVM for an unseen label $c \in C^U$ is estimated by a weighted combination of the parameters of $|C^{Tr}|$ trained SVMs corresponding to known actions.

Graph Convolutional Network (GCN): GCN has been used in ZSL by Lee et al. (2018) and Wang, Ye, and Gupta (2018). We follow the method in Wang et al. (2018) and incorporate GCN into our multi-label ZSL framework. A graph is constructed for label representations with k Nearest Neighbour ($k = 3$ and 6 for Breakfast and Charades datasets respectively). A GCN with four convolutional layers is trained with classifier parameters for known classes, then used to learn the classifier parameters for unseen classes. We use SVMs as the classifiers whose parameters for known classes are obtained in the same way as in COSTA.

Fast0Tag: Fast0Tag is a method for multi-label image tagging and multi-label ZSL (Zhang, Gong, and Shah, 2016). The main idea behind Fast0Tag is learning a mapping function $\phi : X \rightarrow S$ from visual to semantic space for multi-label zero-shot tagging and recognition. Unlike DSP, a ranking-based loss function, *RankNet*, is used to train a deep network to carry out $\phi(\cdot)$ so that for a video clip, its relevant labels should be ranked ahead of those irrelevant ones.

Fast0Tag+: Our work presented in this paper suggests that the use of learned semantic embedding leads to better performance than the use of the original semantic representations directly. To further investigate this idea, we make an extension of Fast0Tag by incorporating our semantic model into the Fast0Tag model and name our extension *Fast0Tag+*. As a result, Fast0Tag+has an architecture resembling ours (c.f. the left box of Fig. 1), where the visual model is carried out by the original Fast0Tag architecture while the semantic model is the same as ours presented in Section 3. The original rank loss functions in Fast0Tag are used and our alternate learning algorithm described in Algorithm 1 is used for parameter estimation. For recognition, the same procedure presented in Section 3.3 is used for a given test instance. Here, we argue that this extension would provide further evidence in examining the effectiveness of semantic embedding learning.

4.6.3. Pooling strategy

To investigate the effect of different pooling strategies over temporal relatedness scores, we conduct a comparative experiment by replacing the average pooling with either the maximum pooling or the local average global maximum pooling in our framework. For the maximum pooling, Eq. (4) for the average pooling is thus altered to

$$\boldsymbol{o}_i = \max_{t=1}^{T} \boldsymbol{o}_{it}. \tag{22}$$

For the local average global maximum pooling, we firstly divide the T segments into T_s groups with a 50% overlap between two consecutive groups. As a result, there are $N_g = 2 * T/T_s$ consecutive segments in each group. We calculate the average score in each group and find the maximum as follows:

$$\boldsymbol{o}_i = \max_{t_s=1}^{T_s} \frac{1}{N_g} \sum_{t=(t_s-1)N_g/2+1}^{(t_s+1)N_g/2} \boldsymbol{o}_{it}. \tag{23}$$

Note that the local average global maximum pooling strategy is generic, and the average pooling and maximum pooling can be viewed as its special cases without between-group overlapping: the average pooling when $T_s = 1, N_g = T$ and the maximum pooling when $T_s = T, N_g = 1$, respectively. To make a thorough investigation, we set $T_s = 10, N_g = 60$ and $T_s = 20, N_g = 30$ as two experimental settings for Breakfast dataset. For Charades dataset, we set $T_s = 5, N_g = 8$ and $T_s = 10, N_g = 4$. All other experimental settings are kept the same for a fair comparison.

In our comparative study, the optimal hyper-parameters involved in baseline and state-of-the-art learning models are sought during their learning with the same cross-validation procedure as described in Section 4.3. Moreover, five state-of-the-art methods described above and ours are extensible to multi-label recognition straightforwardly; i.e., all the actions are known in advance and their training examples are available during learning. Thus, we also report the multi-label recognition performance, which not only extends our comparative study in a wider scope but also provides a benchmark to see how much the performance of each method is degraded in a zero-shot circumstance. For experiments in comparison of different pooling strategies, all the components and setting are kept unchanged except the pooling operations.

5. Experimental results

In this section, we report the detailed experimental results in different settings and exemplify some typical test instances via visual inspection.

5.1. Results on learning

We first report the experimental results regarding learning including optimal hyper-parameters for all the models used in our experiments and the evolution of the learning process for our model trained with our proposed alternate learning algorithm (Algorithm 1) under different data split settings.

As described in Section 4.3, we employ a grid-based search procedure via cross-validation to find out the optimal hyper-parameters in terms of both the loss used to train a model and the I-MAP performance (c.f. Section 4.5) as this metric directly evaluates the relatedness of a video instance to all the labels in a considered action label collection. We seek an optimal value from a set of candidate hyper-parameters involved in all different learning models used in our experiments with the exactly same procedure as follows:

Network architecture: The optimal architecture of neural networks in a model used in our experiments is investigated by tuning different number of neurons in each hidden layer. In our proposed model, there are totally four structural hyper-parameters. The number of hidden units in the LSTM layer is selected from the candidate set, $N_1 = 256, 512, 1024$. In the visual model, the number of neurons in the hidden layer above the LSTM layer is investigated with $N_2 = 1024, 2048$. In the semantic model, the number of neurons in the first hidden layer is selected from $N_1 = 300, 500, 700$. As a critical hyper-parameter in our algorithm, the dimension of latent embedding space d_e, the number of neurons in the visual/semantic embedding layers, is investigated by setting the candidate values, $d_e = 200, 500, 800$. For the *non-recurrent* baseline model, the number of neurons in the first hidden layer replacing the LSTM layer is chosen from $N_1 = 1024, 2048, 4096$. For Fast0Tag and Fast0Tag+, the number of neurons in the first and second hidden layers are selected from $N_1 = 2048, 4096, 8092$ and $N_2 = 1024, 2048$, respectively.

Learning rate: For all the neural networks in the proposed model, the baseline and the state-of-the-art models, candidate learning rates are $\{1e{-}2, 1e{-}4\}$ and $\{1e{-}4, 1e{-}6\}$ for the visual and the semantic models, respectively.

Number of epochs: Learning is stopped when the I-MAP performance on a validation set is no longer improved within the last 10 epochs and the loss reaches a low level on both training and validation sets. Then, the optimal model chosen is the one that yields the highest value of I-MAP on the validation set.

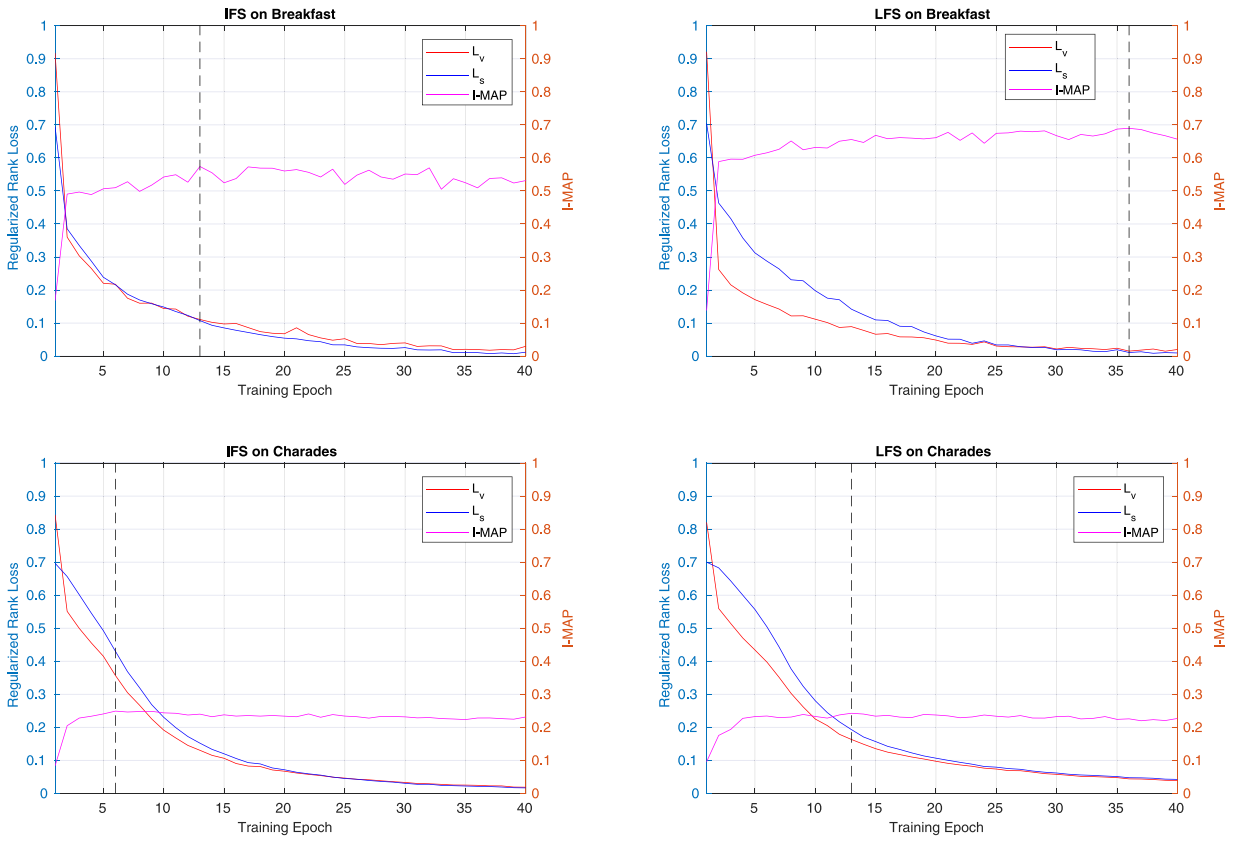

Fig. 3. The evolution of the regularized RankNet losses, L_v and L_s, on training data and the I-MAP values on validation data during the joint visual and semantic embedding learning with our alternate learning algorithm described in Algorithm 1. All the results are achieved based on Split 1 on two datasets, Breakfast and Charades, under the IFS and the LFS settings (c.f. Table 2).

Dropout rate: The dropout rate used in the first layer of a neural model during learning is selected from {0, 0.5}.
Margin: The margin used in the hinge rank loss is selected for $m = 0.1, 1, 10$.
SVM hyper-parameters: In our comparative study, ConSE (Norouzi et al., 2014) and COSTA (Mensink et al., 2014) employ a linear SVM for classification and DSP (Lampert et al., 2014) uses a linear SVR for regression. In our experiments, an optimal soft-margin value is sought from $C = 0.01, 1, 100$. For SVR, the percentage of support vectors is always set to $\epsilon = 0.1$ as suggested in literature.

As a result, the resultant optimal hyper-parameter values in different experimental settings are summarized in Table 7.

To train our model described in Section 3, our proposed learning algorithm optimizes two rank loss functions alternately for joint visual and semantic embedding learning. With the regularized RankNet loss functions, we would exhibit the learning behaviour during the training. As illustrated in Fig. 3, the regularized rank losses, L_v and L_s, with respect to the visual and the semantic models keep decreasing steadily on training data as the training epochs increase regardless of the data split settings and datasets. Nevertheless, we adopt the early-stop strategy to avoid overfitting. However, we observe that the change of two ranking losses on validation data fluctuates wildly in learning so that we cannot decide a proper early-stop point easily. Instead we use the I-MAP measured on validation data to decide the proper early-stop points, as shown in Fig. 3 where the bars of dash line indicate the actual point that the learning is stopped for different training datasets. In general, all our experiments in learning (including not shown in Fig. 3) suggest that our alternate learning algorithm always converges regardless of different rank losses and datasets under different data split settings.

5.2. Results on comparison to baseline models

Tables 3 and 4 summarize all the results yielded by four baseline models described in Section 4.6.1 and the full model described in Section 3.2, with the use of regularized RankNet loss and hinge rank loss described in Section 3.2.2 respectively. The experimental results are reported based on two different data split settings described in Section 4.1.2, *instance-first split* (IFS) and *label-first split* (LFS), under three different evaluation scenarios described in Section 4.4; i.e., *generalized ZSL*, *known-action only* and *unseen-action only* scenarios. For reliability, we report the *mean* and *standard error of the mean* (SEM) of results ($k = 5$ used in evaluation metrics, i.e., Eqs. (19)–(21)) over three randomly generated known/unseen label splits for each evaluation scenario.

For the IFS setting, it is observed from Tables 3 and 4 that all the baseline models and the full model perform significantly better than the RGS, a random guess model, on two datasets regardless of evaluation scenarios apart from the RLR model under the unseen-action only scenario. Due to a lack of knowledge transfer in a random label representation, the zero-shot performance of the RLR is expected. Overall, the full model outperforms all the baseline models on both datasets in the generalized ZSL and unseen-action only scenarios regardless of evaluation metrics. A comparison to the WSE suggests that the performance of the full model is generally superior to this baseline on both datasets under different evaluation scenarios, which lends evidence to support the necessity of the semantic embedding learning in multi-label learning problems. Also, we observe that the full model outperforms the RLR on Breakfast but fails to do so on Charades in the know-action only scenario when using RankNet

Table 3
Multi-label zero-shot recognition performance (mean±SEM%) of the baseline models and the full model using the *RankNet* loss in three evaluation scenarios under different data split settings. **Notation**: IFS — Instance-First Split; LFS — Label-First Split; GZSL — generalized ZSL scenario; KnownA — Known-action only scenario; UnseenA — Unseen-action only scenario.

Data split	Evaluation scenario	Model	Breakfast					Charades				
			L-MAP	I-MAP	P	R	F_1	L-MAP	I-MAP	P	R	F_1
IFS	GZSL	RGS	10.9 ± 0.0	15.4 ± 0.0	8.8 ± 0.0	10.2 ± 0.0	9.4 ± 0.0	5.9 ± 0.0	8.5 ± 0.0	5.6 ± 0.0	3.2 ± 0.0	4.1 ± 0.0
		NRC	27.9 ± 0.7	49.1 ± 1.1	35.1 ± 0.9	40.5 ± 1.1	37.6 ± 1.0	9.2 ± 0.1	20.9 ± 0.7	20.7 ± 0.9	11.9 ± 0.5	15.1 ± 0.7
		WSE	30.2 ± 0.4	50.1 ± 0.4	36.7 ± 0.6	42.4 ± 0.7	39.3 ± 0.6	9.3 ± 0.1	20.6 ± 0.4	20.5 ± 0.5	11.7 ± 0.3	14.9 ± 0.3
		RLR	29.6 ± 0.3	50.4 ± 0.5	36.7 ± 0.7	42.3 ± 0.8	39.3 ± 0.8	9.4 ± 0.1	20.7 ± 1.0	21.1 ± 1.4	12.0 ± 0.8	15.3 ± 1.0
		Ours	**32.8 ± 0.7**	**53.5 ± 1.2**	**38.6 ± 1.6**	**44.5 ± 1.9**	**41.4 ± 1.8**	**9.7 ± 0.1**	**22.4 ± 0.4**	**22.7 ± 0.4**	**13.0 ± 0.2**	**16.5 ± 0.3**
	KnownA	RGS	11.4 ± 0.2	17.4 ± 0.1	9.6 ± 0.2	12.9 ± 0.0	11.0 ± 0.1	6.1 ± 0.1	9.4 ± 0.1	5.8 ± 0.1	4.3 ± 0.0	4.9 ± 0.0
		NRC	30.3 ± 1.0	53.6 ± 0.8	35.4 ± 0.7	47.5 ± 0.6	40.6 ± 0.6	10.0 ± 0.2	25.4 ± 0.5	23.0 ± 0.6	17.1 ± 0.2	19.6 ± 0.4
		WSE	32.1 ± 0.9	55.4 ± 1.0	37.6 ± 0.3	50.6 ± 0.5	43.2 ± 0.2	10.1 ± 0.3	24.6 ± 0.3	22.3 ± 0.3	16.6 ± 0.0	19.0 ± 0.1
		RLR	33.5 ± 1.1	56.3 ± 0.8	37.4 ± 1.0	50.2 ± 0.8	42.8 ± 0.9	**10.7 ± 0.1**	**26.7 ± 0.7**	**23.9 ± 0.9**	**17.8 ± 0.4**	**20.4 ± 0.6**
		Ours	**35.3 ± 1.3**	**58.0 ± 0.4**	**38.2 ± 1.6**	**51.3 ± 1.4**	**43.8 ± 1.5**	10.5 ± 0.2	26.1 ± 0.3	23.5 ± 0.5	17.5 ± 0.1	20.0 ± 0.2
	UnseenA	RGS	8.5 ± 0.6	30.7 ± 0.6	6.1 ± 0.6	50.0 ± 0.0	10.9 ± 0.9	5.4 ± 0.0	13.9 ± 0.0	5.1 ± 0.0	12.5 ± 0.0	7.2 ± 0.0
		NRC	17.1 ± 0.7	47.5 ± 2.4	8.7 ± 0.8	70.8 ± 2.5	15.4 ± 1.2	6.8 ± 0.4	20.4 ± 1.1	8.4 ± 0.7	20.8 ± 1.9	11.9 ± 1.0
		WSE	21.7 ± 1.7	47.7 ± 3.0	9.1 ± 1.9	72.4 ± 8.8	16.1 ± 3.2	6.9 ± 0.4	20.5 ± 1.6	8.5 ± 0.7	21.2 ± 2.1	12.1 ± 1.0
		RLR	13.2 ± 2.5	34.3 ± 7.1	7.0 ± 1.6	56.6 ± 9.7	12.4 ± 2.7	5.5 ± 0.4	13.4 ± 0.6	4.7 ± 0.4	11.6 ± 0.6	6.7 ± 0.5
		Ours	**22.3 ± 1.1**	**53.1 ± 5.8**	**9.5 ± 1.5**	**77.2 ± 8.4**	**16.9 ± 2.5**	**7.1 ± 0.4**	**22.4 ± 2.1**	**9.5 ± 1.0**	**23.5 ± 2.7**	**13.5 ± 1.5**
LFS	GZSL	RGS	15.2 ± 1.5	17.8 ± 0.4	11.3 ± 0.4	10.2 ± 0.0	10.7 ± 0.2	5.2 ± 0.1	7.9 ± 0.1	5.1 ± 0.1	3.2 ± 0.0	3.9 ± 0.0
		NRC	21.9 ± 2.2	27.6 ± 1.2	19.2 ± 1.0	17.3 ± 1.1	18.2 ± 1.0	8.9 ± 0.2	20.5 ± 0.5	21.2 ± 0.8	13.3 ± 0.5	16.4 ± 0.6
		WSE	**25.2 ± 1.4**	31.0 ± 2.7	23.0 ± 2.0	20.7 ± 1.4	21.7 ± 1.6	9.1 ± 0.0	20.3 ± 0.0	21.0 ± 0.2	13.2 ± 0.2	16.2 ± 0.2
		RLR	22.6 ± 2.1	29.6 ± 1.9	22.0 ± 0.7	19.8 ± 0.2	20.9 ± 0.3	**9.3 ± 0.2**	19.6 ± 0.4	19.8 ± 0.6	12.4 ± 0.4	15.2 ± 0.5
		Ours	25.0 ± 1.4	**32.6 ± 2.9**	**23.6 ± 2.1**	**21.2 ± 1.5**	**22.3 ± 1.7**	9.2 ± 0.1	**20.8 ± 0.3**	**21.6 ± 0.7**	**13.5 ± 0.4**	**16.6 ± 0.5**
	KnownA	RGS	14.7 ± 1.9	17.8 ± 0.2	10.1 ± 0.2	12.8 ± 0.0	11.3 ± 0.1	5.0 ± 0.1	8.0 ± 0.1	4.8 ± 0.1	3.7 ± 0.0	4.2 ± 0.0
		NRC	22.2 ± 2.5	30.2 ± 1.5	18.9 ± 1.1	24.2 ± 1.8	21.2 ± 1.4	8.9 ± 0.3	22.3 ± 0.6	21.1 ± 0.7	15.9 ± 0.4	18.1 ± 0.5
		WSE	**25.3 ± 2.0**	34.2 ± 3.3	23.1 ± 1.9	29.5 ± 2.7	25.9 ± 2.3	9.0 ± 0.0	22.2 ± 0.3	21.1 ± 0.4	15.9 ± 0.0	18.1 ± 0.1
		RLR	24.8 ± 2.6	34.3 ± 3.1	21.7 ± 0.7	27.6 ± 0.4	24.3 ± 0.6	**9.5 ± 0.1**	**23.0 ± 0.4**	21.2 ± 0.5	16.0 ± 0.4	18.2 ± 0.4
		Ours	25.2 ± 1.6	**35.7 ± 3.6**	**23.3 ± 2.0**	**29.6 ± 2.8**	**26.0 ± 2.3**	9.1 ± 0.1	**23.0 ± 0.6**	**21.6 ± 0.7**	**16.3 ± 0.3**	**18.6 ± 0.5**
	UnseenA	RGS	16.8 ± 1.6	34.2 ± 1.3	16.2 ± 1.6	50.0 ± 0.0	24.4 ± 1.8	6.9 ± 0.1	19.4 ± 0.1	6.7 ± 0.1	25.0 ± 0.0	10.5 ± 0.1
		NRC	21.2 ± 1.4	42.3 ± 5.4	18.7 ± 3.2	57.1 ± 7.0	28.0 ± 4.4	9.4 ± 0.1	**30.7 ± 2.2**	**11.6 ± 0.7**	**43.2 ± 2.8**	**18.2 ± 1.1**
		WSE	**25.2 ± 0.4**	42.5 ± 4.6	20.0 ± 3.1	61.5 ± 6.8	30.1 ± 4.3	**9.7 ± 0.1**	30.1 ± 1.2	11.5 ± 0.4	43.1 ± 1.8	**18.2 ± 0.6**
		RLR	16.7 ± 2.5	32.2 ± 2.1	15.0 ± 1.9	45.8 ± 2.0	22.5 ± 2.4	7.7 ± 0.5	19.3 ± 0.3	6.7 ± 0.1	25.2 ± 0.6	10.6 ± 0.2
		Ours	24.6 ± 1.5	**44.8 ± 4.7**	**22.2 ± 2.5**	**68.6 ± 4.7**	**33.4 ± 3.3**	**9.7 ± 0.1**	29.2 ± 2.1	11.1 ± 0.9	41.6 ± 3.5	17.5 ± 1.4

Table 4
Multi-label zero-shot recognition performance (mean±SEM%) of the baseline models and our full model using the *Hinge rank* loss in three evaluation scenarios under different data split settings. Notations are the same as described in Table 3.

Data split	Evaluation scenario	Model	Breakfast					Charades				
			L-MAP	I-MAP	P	R	F_1	L-MAP	I-MAP	P	R	F_1
IFS	GZSL	RGS	10.9 ± 0.0	15.4 ± 0.0	8.8 ± 0.0	10.2 ± 0.0	9.4 ± 0.0	5.9 ± 0.0	8.5 ± 0.0	5.6 ± 0.0	3.2 ± 0.0	4.1 ± 0.0
		NRC	28.1 ± 0.4	50.1 ± 1.0	36.7 ± 0.7	42.3 ± 0.8	39.3 ± 0.7	9.2 ± 0.1	21.7 ± 0.7	22.2 ± 0.7	12.7 ± 0.4	16.2 ± 0.5
		WSE	31.1 ± 0.7	50.4 ± 0.3	36.6 ± 0.5	42.2 ± 0.6	39.2 ± 0.6	8.6 ± 0.0	20.1 ± 0.5	20.3 ± 0.7	11.6 ± 0.4	14.7 ± 0.5
		RLR	30.3 ± 1.2	51.8 ± 1.8	37.4 ± 1.4	43.2 ± 1.6	40.1 ± 1.5	9.5 ± 0.1	21.4 ± 0.9	22.4 ± 0.8	12.8 ± 0.5	16.3 ± 0.6
		Ours	**32.7 ± 0.4**	**53.4 ± 0.8**	**38.9 ± 0.5**	**44.9 ± 0.5**	**41.7 ± 0.5**	**10.0 ± 0.1**	**22.6 ± 0.4**	**23.1 ± 0.5**	**13.2 ± 0.3**	**16.8 ± 0.4**
	KnownA	RGS	11.4 ± 0.2	17.4 ± 0.1	9.6 ± 0.2	12.9 ± 0.0	11.0 ± 0.1	6.1 ± 0.1	9.4 ± 0.1	5.8 ± 0.1	4.3 ± 0.0	4.9 ± 0.0
		NRC	30.5 ± 0.6	54.9 ± 0.9	36.6 ± 0.4	49.3 ± 0.4	42.0 ± 0.2	10.1 ± 0.3	25.4 ± 0.5	23.2 ± 0.6	17.2 ± 0.2	19.8 ± 0.4
		WSE	33.0 ± 0.9	55.4 ± 0.7	37.1 ± 0.6	49.9 ± 0.6	42.6 ± 0.5	9.3 ± 0.2	23.7 ± 0.4	21.5 ± 0.5	16.0 ± 0.2	18.3 ± 0.3
		RLR	34.9 ± 1.8	**59.0 ± 1.3**	**38.6 ± 0.7**	51.9 ± 1.1	**44.3 ± 0.8**	10.8 ± 0.2	**26.5 ± 0.7**	23.7 ± 0.8	17.7 ± 0.4	20.3 ± 0.5
		Ours	**35.6 ± 0.5**	58.2 ± 1.0	**38.6 ± 0.3**	**52.0 ± 0.5**	**44.3 ± 0.0**	**10.9 ± 0.2**	26.4 ± 0.3	**24.1 ± 0.3**	**17.9 ± 0.2**	**20.6 ± 0.1**
	UnseenA	RGS	8.5 ± 0.6	30.7 ± 0.6	6.1 ± 0.6	50.0 ± 0.0	10.9 ± 0.9	5.4 ± 0.0	13.9 ± 0.0	5.1 ± 0.0	12.5 ± 0.0	7.2 ± 0.0
		NRC	17.6 ± 1.6	46.7 ± 3.2	**9.1 ± 1.2**	**73.4 ± 4.0**	**16.1 ± 2.0**	6.7 ± 0.4	21.8 ± 1.2	9.3 ± 0.4	23.1 ± 1.4	13.2 ± 0.6
		WSE	**22.7 ± 2.1**	43.6 ± 6.0	**9.1 ± 1.9**	72.6 ± 8.3	**16.1 ± 3.1**	6.6 ± 0.4	21.3 ± 1.9	8.7 ± 0.7	21.7 ± 2.6	12.4 ± 1.1
		RLR	10.5 ± 2.0	35.5 ± 2.1	5.9 ± 0.3	48.4 ± 2.1	10.5 ± 0.5	5.6 ± 0.3	14.7 ± 1.9	5.0 ± 0.7	12.4 ± 2.0	7.1 ± 1.1
		Ours	20.3 ± 0.6	**47.6 ± 1.6**	8.8 ± 1.0	71.6 ± 2.2	15.7 ± 1.6	**7.3 ± 0.5**	**22.6 ± 1.5**	**9.6 ± 0.7**	**23.9 ± 2.0**	**13.7 ± 1.0**
LFS	GZSL	RGS	15.2 ± 1.5	17.8 ± 0.4	11.3 ± 0.4	10.2 ± 0.0	10.7 ± 0.2	5.2 ± 0.1	7.9 ± 0.1	5.1 ± 0.1	3.2 ± 0.0	3.9 ± 0.0
		NRC	22.6 ± 2.0	26.6 ± 1.7	18.5 ± 1.3	16.7 ± 1.2	17.5 ± 1.2	8.9 ± 0.1	20.7 ± 0.1	21.2 ± 0.3	13.3 ± 0.3	16.4 ± 0.3
		WSE	24.4 ± 1.7	31.3 ± 3.0	23.4 ± 1.9	21.1 ± 1.4	22.2 ± 1.6	8.1 ± 0.2	19.4 ± 0.2	19.4 ± 0.4	12.2 ± 0.3	14.9 ± 0.3
		RLR	22.2 ± 1.4	31.6 ± 1.6	24.3 ± 2.1	21.8 ± 1.6	23.0 ± 1.8	9.0 ± 0.1	19.9 ± 0.7	20.7 ± 1.0	13.0 ± 0.5	16.0 ± 0.6
		Ours	**24.7 ± 1.4**	**32.9 ± 2.9**	**24.6 ± 2.5**	**22.1 ± 1.9**	**23.3 ± 2.2**	**9.2 ± 0.1**	**21.1 ± 0.4**	**22.1 ± 0.7**	**13.9 ± 0.4**	**17.1 ± 0.5**
	KnownA	RGS	14.7 ± 1.9	17.8 ± 0.2	10.1 ± 0.2	12.8 ± 0.0	11.3 ± 0.1	5.0 ± 0.1	8.0 ± 0.1	4.8 ± 0.1	3.7 ± 0.0	4.2 ± 0.0
		NRC	23.0 ± 2.6	29.4 ± 2.7	18.5 ± 1.5	23.6 ± 2.2	20.7 ± 1.8	8.8 ± 0.1	22.6 ± 0.4	21.3 ± 0.4	16.1 ± 0.0	18.3 ± 0.1
		WSE	24.9 ± 2.4	35.1 ± 3.8	23.4 ± 2.0	29.7 ± 2.0	26.2 ± 2.0	7.9 ± 0.2	21.1 ± 0.5	19.6 ± 0.4	14.8 ± 0.1	16.9 ± 0.2
		RLR	23.6 ± 2.6	36.1 ± 2.7	24.3 ± 2.2	31.0 ± 3.2	27.2 ± 2.6	**9.2 ± 0.1**	22.7 ± 0.9	21.0 ± 1.0	15.9 ± 0.5	18.1 ± 0.7
		Ours	**25.2 ± 1.8**	**37.1 ± 3.4**	**25.0 ± 2.0**	**31.9 ± 2.7**	**28.1 ± 2.3**	9.1 ± 0.0	**23.2 ± 0.4**	**22.1 ± 0.5**	**16.6 ± 0.2**	**19.0 ± 0.3**
	UnseenA	RGS	16.8 ± 1.6	34.2 ± 1.3	16.2 ± 1.6	50.0 ± 0.0	24.4 ± 1.8	6.9 ± 0.1	19.4 ± 0.1	6.7 ± 0.1	25.0 ± 0.0	10.5 ± 0.1
		NRC	21.6 ± 0.6	39.6 ± 2.8	**19.7 ± 1.4**	**61.4 ± 4.6**	**29.7 ± 1.8**	9.5 ± 0.2	30.7 ± 2.3	11.8 ± 0.7	44.0 ± 2.8	18.6 ± 1.1
		WSE	**23.3 ± 0.2**	**41.2 ± 4.3**	19.0 ± 2.8	58.4 ± 5.7	28.6 ± 3.8	9.3 ± 0.1	29.9 ± 1.6	11.5 ± 0.2	42.9 ± 1.3	18.1 ± 0.4
		RLR	18.9 ± 1.1	33.3 ± 1.2	16.4 ± 0.1	51.5 ± 5.4	24.7 ± 0.6	7.4 ± 0.5	18.5 ± 0.8	6.4 ± 0.5	23.8 ± 1.4	10.1 ± 0.7
		Ours	**23.3 ± 2.0**	40.2 ± 5.1	19.4 ± 3.6	59.0 ± 7.7	29.1 ± 5.0	**10.0 ± 0.1**	**30.9 ± 2.6**	**12.0 ± 0.9**	**44.9 ± 3.8**	**18.9 ± 1.5**

Table 5
Multi-label zero-shot recognition performance (mean±SEM%) of six state-of-the-art methods and ours with the reference to random guess in three evaluation scenarios under different data split settings. † denotes our adaptation to multi-label ZSL. The notations are the same as used in Table 3.

Data split	Evaluation scenario	Model	Breakfast					Charades				
			L-MAP	I-MAP	P	R	F_1	L-MAP	I-MAP	P	R	F_1
IFS	GZSL	RGS	10.9 ± 0.0	15.4 ± 0.0	8.8 ± 0.0	10.2 ± 0.0	9.4 ± 0.0	5.9 ± 0.0	8.5 ± 0.0	5.6 ± 0.0	3.2 ± 0.0	4.1 ± 0.0
		DSP (Lampert et al., 2014)	21.3 ± 0.0	25.4 ± 0.9	16.8 ± 0.4	19.4 ± 0.4	18.0 ± 0.4	7.9 ± 0.0	12.5 ± 0.1	12.5 ± 0.2	7.2 ± 0.1	9.1 ± 0.1
		ConSE (Norouzi et al., 2014)	16.1 ± 0.2	29.6 ± 0.3	20.3 ± 0.1	23.4 ± 0.1	21.7 ± 0.1	7.3 ± 0.0	13.9 ± 0.3	14.6 ± 0.5	8.3 ± 0.3	10.6 ± 0.4
		COSTA (Mensink et al., 2014)	19.7 ± 0.2	37.4 ± 0.3	28.8 ± 0.4	33.2 ± 0.5	30.8 ± 0.5	8.5 ± 0.1	16.9 ± 0.7	17.6 ± 0.9	10.1 ± 0.5	12.8 ± 0.6
		†GCN (Wang et al., 2018)	20.0 ± 0.2	36.6 ± 0.4	27.9 ± 0.4	32.2 ± 0.5	29.9 ± 0.4	8.5 ± 0.1	16.6 ± 0.5	17.2 ± 0.8	9.8 ± 0.5	12.5 ± 0.6
		Fast0Tag (Zhang, Gong, and Shah, 2016)	22.3 ± 1.1	38.6 ± 0.2	27.5 ± 0.1	31.8 ± 0.1	29.5 ± 0.1	9.3 ± 0.0	20.6 ± 1.0	20.6 ± 1.4	11.8 ± 0.8	15.0 ± 1.0
		Fast0Tag+	23.4 ± 0.3	40.6 ± 0.7	29.2 ± 0.3	33.7 ± 0.3	31.3 ± 0.3	9.7 ± 0.1	21.7 ± 0.6	21.9 ± 0.8	12.5 ± 0.5	15.9 ± 0.6
		Ours (RankNet)	32.1 ± 0.8	53.3 ± 1.0	39.0 ± 0.9	45.0 ± 1.0	41.8 ± 0.9	9.7 ± 0.1	22.4 ± 0.4	22.8 ± 0.4	13.0 ± 0.2	16.5 ± 0.3
		Ours (Hinge)	32.7 ± 0.4	53.4 ± 0.8	38.9 ± 0.5	44.9 ± 0.5	41.7 ± 0.5	10.0 ± 0.1	22.6 ± 0.4	23.1 ± 0.5	13.2 ± 0.3	16.8 ± 0.4
		Ours (Fusion)	**33.9 ± 0.4**	**54.7 ± 1.1**	**40.0 ± 1.1**	**46.1 ± 1.3**	**42.8 ± 1.2**	**10.1 ± 0.1**	**23.3 ± 0.5**	**23.7 ± 0.7**	**13.5 ± 0.4**	**17.2 ± 0.5**
	KnownA	RGS	11.4 ± 0.2	17.4 ± 0.1	9.6 ± 0.2	12.9 ± 0.0	11.0 ± 0.1	6.1 ± 0.1	9.4 ± 0.1	5.8 ± 0.1	4.3 ± 0.0	4.9 ± 0.0
		DSP (Lampert et al., 2014)	22.6 ± 0.5	30.8 ± 1.9	19.1 ± 0.9	25.7 ± 1.3	21.9 ± 1.1	8.5 ± 0.1	13.9 ± 0.3	12.4 ± 0.4	9.2 ± 0.3	10.6 ± 0.3
		ConSE (Norouzi et al., 2014)	17.1 ± 0.3	33.0 ± 1.2	21.0 ± 0.2	28.2 ± 0.7	24.1 ± 0.4	7.8 ± 0.2	15.9 ± 0.4	14.6 ± 0.7	10.8 ± 0.3	12.4 ± 0.5
		COSTA (Mensink et al., 2014)	22.1 ± 0.7	41.6 ± 0.7	28.8 ± 0.4	38.7 ± 0.2	33.0 ± 0.3	9.3 ± 0.2	19.7 ± 0.6	17.6 ± 0.9	13.1 ± 0.5	15.0 ± 0.6
		†GCN (Wang et al., 2018)	22.1 ± 0.7	41.6 ± 0.7	28.8 ± 0.4	38.7 ± 0.2	33.0 ± 0.3	9.3 ± 0.2	19.7 ± 0.6	17.6 ± 0.9	13.1 ± 0.5	15.0 ± 0.6
		Fast0Tag (Zhang, Gong, and Shah, 2016)	23.9 ± 1.3	44.3 ± 0.5	29.8 ± 0.7	40.0 ± 0.6	34.1 ± 0.6	10.0 ± 0.2	24.6 ± 0.6	22.4 ± 0.7	16.6 ± 0.3	19.1 ± 0.4
		Fast0Tag+	25.4 ± 0.2	45.0 ± 0.5	29.9 ± 0.4	40.2 ± 1.1	34.3 ± 0.6	10.5 ± 0.3	25.7 ± 0.7	23.3 ± 0.9	17.3 ± 0.4	19.8 ± 0.6
		Ours (RankNet)	34.5 ± 1.0	57.8 ± 0.7	38.5 ± 1.0	51.8 ± 0.6	44.2 ± 0.9	10.5 ± 0.2	26.1 ± 0.3	23.5 ± 0.5	17.5 ± 0.1	20.0 ± 0.2
		Ours (Hinge)	35.6 ± 0.5	58.2 ± 1.0	38.6 ± 0.3	52.0 ± 0.5	44.3 ± 0.0	10.9 ± 0.2	26.4 ± 0.3	24.1 ± 0.3	17.9 ± 0.2	20.6 ± 0.1
		Ours (Fusion)	**36.6 ± 0.7**	**59.4 ± 0.5**	**39.6 ± 0.9**	**53.2 ± 0.5**	**45.4 ± 0.8**	**11.0 ± 0.3**	**27.1 ± 0.4**	**24.6 ± 0.5**	**18.3 ± 0.2**	**21.0 ± 0.3**
	UnseenA	RGS	8.5 ± 0.6	30.7 ± 0.6	6.1 ± 0.6	50.0 ± 0.0	10.9 ± 0.9	5.4 ± 0.0	13.9 ± 0.0	5.1 ± 0.0	12.5 ± 0.0	7.2 ± 0.0
		DSP (Lampert et al., 2014)	15.9 ± 1.5	27.0 ± 4.7	5.8 ± 1.6	47.6 ± 11.5	10.4 ± 2.7	6.2 ± 0.4	17.3 ± 1.4	7.4 ± 0.8	18.4 ± 1.9	10.6 ± 1.1
		ConSE (Norouzi et al., 2014)	12.2 ± 0.3	29.9 ± 5.0	6.2 ± 0.9	51.0 ± 6.9	11.1 ± 1.5	5.8 ± 0.4	17.3 ± 0.8	7.0 ± 0.7	17.2 ± 1.1	9.9 ± 0.8
		COSTA (Mensink et al., 2014)	9.2 ± 1.2	37.4 ± 2.8	7.4 ± 0.7	60.1 ± 2.5	13.1 ± 1.0	6.0 ± 0.3	15.5 ± 1.0	6.3 ± 0.8	15.4 ± 1.5	8.9 ± 1.0
		†GCN (Wang et al., 2018)	11.0 ± 1.4	29.9 ± 4.6	5.8 ± 1.4	47.3 ± 10.3	10.3 ± 2.4	5.9 ± 0.4	15.3 ± 1.3	6.2 ± 1.5	15.0 ± 3.1	8.7 ± 2.0
		Fast0Tag (Zhang, Gong, and Shah, 2016)	15.3 ± 0.9	36.7 ± 4.1	7.0 ± 1.4	55.9 ± 6.7	12.4 ± 2.3	7.1 ± 0.4	20.2 ± 2.4	8.3 ± 0.8	20.8 ± 2.7	11.9 ± 1.2
		Fast0Tag+	15.1 ± 1.2	39.4 ± 1.3	7.4 ± 0.9	60.1 ± 3.6	13.2 ± 1.5	**7.3 ± 0.4**	19.3 ± 0.5	8.1 ± 0.3	20.0 ± 0.6	11.5 ± 0.3
		Ours (RankNet)	21.9 ± 0.3	51.0 ± 4.5	9.4 ± 1.2	76.5 ± 6.8	16.7 ± 2.0	7.1 ± 0.4	22.4 ± 2.1	9.5 ± 1.0	23.5 ± 2.7	13.5 ± 1.5
		Ours (Hinge)	20.3 ± 0.6	47.6 ± 1.6	8.8 ± 1.0	71.6 ± 2.2	15.7 ± 1.6	**7.3 ± 0.5**	22.6 ± 1.5	9.6 ± 0.7	23.9 ± 2.0	13.7 ± 1.0
		Ours (Fusion)	**22.3 ± 0.4**	**52.9 ± 4.6**	**9.7 ± 1.6**	**78.8 ± 6.8**	**17.3 ± 2.6**	**7.3 ± 0.5**	**23.1 ± 1.8**	**9.9 ± 0.8**	**24.6 ± 2.5**	**14.1 ± 1.2**
LFS	GZSL	RGS	15.2 ± 1.5	17.8 ± 0.4	11.3 ± 0.4	10.2 ± 0.0	10.7 ± 0.2	5.2 ± 0.1	7.9 ± 0.1	5.1 ± 0.1	3.2 ± 0.0	3.9 ± 0.0
		DSP (Lampert et al., 2014)	20.7 ± 1.7	18.6 ± 1.9	11.0 ± 1.5	9.9 ± 1.0	10.4 ± 1.2	7.4 ± 0.1	12.1 ± 0.3	12.4 ± 0.6	7.7 ± 0.3	9.5 ± 0.4
		ConSE (Norouzi et al., 2014)	18.5 ± 2.1	20.2 ± 1.8	12.7 ± 1.1	11.4 ± 0.6	12.0 ± 0.8	7.0 ± 0.1	13.8 ± 0.1	14.8 ± 0.1	9.3 ± 0.1	11.4 ± 0.1
		COSTA (Mensink et al., 2014)	19.3 ± 2.1	22.7 ± 2.1	16.8 ± 1.0	15.1 ± 0.7	15.9 ± 0.8	8.9 ± 0.1	17.3 ± 0.1	18.6 ± 0.1	11.7 ± 0.2	14.4 ± 0.1
		†GCN (Wang et al., 2018)	19.6 ± 2.2	23.1 ± 2.2	16.9 ± 0.9	15.2 ± 0.6	16.0 ± 0.7	8.9 ± 0.1	17.1 ± 0.1	18.0 ± 0.3	11.3 ± 0.3	13.9 ± 0.3
		Fast0Tag (Zhang, Gong, and Shah, 2016)	22.5 ± 1.5	24.3 ± 1.7	16.2 ± 0.6	14.6 ± 0.1	15.4 ± 0.3	8.6 ± 0.1	20.1 ± 0.4	20.1 ± 0.9	12.6 ± 0.6	15.5 ± 0.7
		Fast0Tag+	21.9 ± 1.1	23.3 ± 1.1	15.3 ± 0.2	13.8 ± 0.6	14.5 ± 0.4	9.0 ± 0.1	20.9 ± 0.3	21.4 ± 0.6	13.5 ± 0.4	16.5 ± 0.5
		Ours (RankNet)	25.0 ± 1.4	32.6 ± 2.9	23.6 ± 2.1	21.2 ± 1.5	22.3 ± 1.7	9.2 ± 0.1	20.8 ± 0.3	21.6 ± 0.7	13.5 ± 0.4	16.6 ± 0.5
		Ours (Hinge)	24.7 ± 1.4	32.9 ± 2.9	**24.6 ± 2.5**	**22.1 ± 1.9**	**23.3 ± 2.2**	9.2 ± 0.1	21.1 ± 0.4	22.1 ± 0.7	13.9 ± 0.4	17.1 ± 0.5
		Ours (Fusion)	**25.5 ± 1.4**	**33.3 ± 2.5**	**24.6 ± 2.3**	**22.1 ± 1.6**	23.2 ± 1.9	**9.6 ± 0.1**	**21.5 ± 0.4**	**22.6 ± 0.6**	**14.2 ± 0.4**	**17.4 ± 0.4**
	KnownA	RGS	14.7 ± 1.9	17.8 ± 0.2	10.1 ± 0.2	12.8 ± 0.0	11.3 ± 0.1	5.0 ± 0.1	8.0 ± 0.1	4.8 ± 0.1	3.7 ± 0.0	4.2 ± 0.0
		DSP (Lampert et al., 2014)	14.1 ± 1.2	18.1 ± 2.7	8.5 ± 1.8	10.3 ± 2.1	9.4 ± 1.9	7.3 ± 0.1	12.9 ± 0.4	12.3 ± 0.7	9.3 ± 0.3	10.6 ± 0.5
		ConSE (Norouzi et al., 2014)	12.2 ± 0.8	22.1 ± 0.6	13.1 ± 0.6	14.6 ± 1.2	13.8 ± 0.9	6.8 ± 0.1	15.0 ± 0.1	14.7 ± 0.2	11.1 ± 0.1	12.6 ± 0.1
		COSTA (Mensink et al., 2014)	19.8 ± 2.9	25.1 ± 2.7	16.8 ± 1.0	21.3 ± 0.8	18.7 ± 0.9	9.0 ± 0.1	19.5 ± 0.1	18.6 ± 0.1	14.1 ± 0.2	16.0 ± 0.1
		†GCN (Wang et al., 2018)	19.8 ± 2.9	25.1 ± 2.7	16.8 ± 1.0	21.3 ± 0.8	18.7 ± 0.9	9.0 ± 0.1	19.5 ± 0.1	18.6 ± 0.1	14.1 ± 0.2	16.0 ± 0.1
		Fast0Tag (Zhang, Gong, and Shah, 2016)	23.1 ± 2.0	26.5 ± 1.9	15.9 ± 0.4	20.2 ± 0.8	17.8 ± 0.5	8.4 ± 0.1	22.4 ± 0.7	20.8 ± 0.7	15.7 ± 0.2	17.9 ± 0.4
		Fast0Tag+	22.2 ± 1.7	24.9 ± 1.5	15.1 ± 0.6	19.3 ± 1.1	17.0 ± 0.8	9.0 ± 0.1	22.8 ± 0.5	21.3 ± 0.5	16.1 ± 0.1	18.3 ± 0.3
		Ours (RankNet)	25.2 ± 1.6	35.7 ± 3.6	23.3 ± 2.0	29.6 ± 2.8	26.0 ± 2.3	9.1 ± 0.1	23.0 ± 0.6	21.6 ± 0.7	16.3 ± 0.3	18.6 ± 0.5
		Ours (Hinge)	25.2 ± 1.8	37.1 ± 3.4	**25.0 ± 2.0**	**31.9 ± 2.7**	**28.1 ± 2.3**	9.1 ± 0.0	23.2 ± 0.4	22.1 ± 0.5	16.6 ± 0.2	19.0 ± 0.3
		Ours (Fusion)	**25.8 ± 1.8**	**37.4 ± 2.7**	24.6 ± 2.0	31.3 ± 2.8	27.6 ± 2.3	**9.5 ± 0.1**	**23.8 ± 0.6**	**22.7 ± 0.7**	**17.1 ± 0.3**	**19.5 ± 0.4**
	UnseenA	RGS	16.8 ± 1.6	34.2 ± 1.3	16.2 ± 1.6	50.0 ± 0.0	24.4 ± 1.8	6.9 ± 0.1	19.4 ± 0.1	6.7 ± 0.1	25.0 ± 0.0	10.5 ± 0.1
		DSP (Lampert et al., 2014)	16.6 ± 0.8	28.8 ± 1.4	14.7 ± 0.6	42.8 ± 1.3	21.9 ± 0.7	8.5 ± 0.2	22.8 ± 0.9	8.1 ± 0.4	30.3 ± 1.3	12.8 ± 0.6
		ConSE (Norouzi et al., 2014)	13.7 ± 1.7	33.0 ± 0.7	14.7 ± 0.5	50.9 ± 4.6	22.7 ± 0.2	7.9 ± 0.4	23.2 ± 0.4	8.5 ± 0.0	31.9 ± 0.3	13.5 ± 0.1
		COSTA (Mensink et al., 2014)	18.2 ± 1.1	35.8 ± 1.2	16.9 ± 1.7	52.2 ± 2.0	25.4 ± 2.0	7.6 ± 0.2	24.9 ± 1.0	9.2 ± 0.4	34.4 ± 1.8	14.5 ± 0.7
		†GCN (Wang et al., 2018)	19.5 ± 1.1	32.6 ± 2.6	15.3 ± 2.6	46.7 ± 4.9	23.0 ± 3.5	7.9 ± 0.2	18.4 ± 0.9	5.8 ± 0.4	21.7 ± 1.5	9.2 ± 0.6
		Fast0Tag (Zhang, Gong, and Shah, 2016)	21.1 ± 1.1	39.1 ± 4.0	18.5 ± 3.6	56.0 ± 6.5	27.8 ± 4.9	9.5 ± 0.1	28.4 ± 2.5	11.0 ± 0.5	41.1 ± 2.2	17.4 ± 0.8
		Fast0Tag+	21.2 ± 0.8	**45.5 ± 3.4**	19.9 ± 1.8	61.7 ± 2.4	30.0 ± 2.1	9.3 ± 0.1	30.2 ± 2.9	11.6 ± 0.7	43.2 ± 3.1	18.2 ± 1.2
		Ours (RankNet)	24.6 ± 1.5	44.8 ± 4.7	**22.2 ± 2.5**	**68.6 ± 4.7**	**33.4 ± 3.3**	9.7 ± 0.1	29.2 ± 2.1	11.1 ± 0.9	41.6 ± 3.5	17.5 ± 1.4
		Ours (Hinge)	23.3 ± 2.0	40.2 ± 5.1	19.4 ± 3.6	59.0 ± 7.7	29.1 ± 5.0	10.0 ± 0.1	30.9 ± 2.6	12.0 ± 0.9	44.9 ± 3.8	18.9 ± 1.5
		Ours (Fusion)	**24.7 ± 1.5**	42.6 ± 6.1	19.6 ± 3.9	59.7 ± 8.6	29.5 ± 5.4	**10.2 ± 0.1**	**31.1 ± 2.7**	**12.2 ± 0.9**	**45.6 ± 3.7**	**19.2 ± 1.4**

loss (Table 3). This result suggests that a semantic representation of labels is not critically important for known actions in multi-label learning when the semantic embedding learning has been employed to explore the between-action relations from label co-occurrences. This observation further implies that the semantic embedding learning cannot explore the semantic relations between labels properly unless there are sufficient training examples for different actions. Nevertheless, the performance in the unseen-action only and the generalized ZSL scenarios clearly indicates the importance of the semantic representation of an action label for knowledge transfer required by ZSL. It is also evident from Tables 3 and 4 that the full model always outperforms the NRC where there are no recurrent connections. Thus, the comparison to the baseline models clearly suggest that the performance gain is brought by the use of an LSTM layer in the visual model and the semantic embedding learning fulfilled in the semantic model.

For the LFS setting, results shown in Tables 3 and 4 suggest that all the baseline models perform significantly better than a random guess. Overall, the full model generally outperforms those baseline models on both datasets. In few circumstances, however, the full model slightly under-performs the WSE on Breakfast in terms of L-MAP with a tiny margin (Table 3). Besides, it is observed from Table 3 that in the unseen-action only scenario, our model slightly under-performs the WSE and the NRC on Charades although it yields the best performance on Breakfast. While from Table 4 we can observe that our full model performs the best on Charades but not on Breakfast. These results reveal that the two employed rank losses are complementary when learning the joint embedding space.

In summary, the comparison to the elaborated baseline models facilitates the understanding of different components and ranking loss functions employed in our proposed framework for multi-label zero-shot human action recognition. Two different ranking losses used in our framework yield the similar performance overall. By comparison to four baseline models, the full model generally leads to better results on two datasets measured with different evaluation metrics in all three evaluation scenarios, although the experimental results also reveal the limitation of components used in the full model to be investigated in our future studies.

5.3. Results on comparison to state-of-the-art methods

Table 5 summarizes the experimental results of the comparative study described in Section 4.6.2. Multi-label ZSL performance of five different methods including Fast0Tag+(our extension for Fast0Tag) with the reference to a random guess is reported to be compared with our proposed framework where two different

Table 6
Multi-label recognition performance (mean±std%) of five state-of-the-art methods and ours.

Model	Breakfast					Charades				
	L-MAP	I-MAP	P	R	F1	L-MAP	I-MAP	P	R	F1
DSP (Lampert et al., 2014)	21.6 ± 0.0	25.2 ± 0.0	16.6 ± 0.0	19.1 ± 0.0	17.7 ± 0.0	8.0 ± 0.0	12.6 ± 0.0	13.0 ± 0.0	7.4 ± 0.0	9.5 ± 0.0
ConSE (Norouzi et al., 2014)	16.7 ± 0.0	30.8 ± 0.0	21.1 ± 0.0	24.4 ± 0.0	22.7 ± 0.0	7.4 ± 0.0	14.6 ± 0.0	15.8 ± 0.0	9.0 ± 0.0	11.5 ± 0.0
COSTA (Mensink et al., 2014)	21.5 ± 0.0	39.3 ± 0.0	29.5 ± 0.0	34.0 ± 0.0	31.6 ± 0.0	9.1 ± 0.0	18.6 ± 0.0	19.5 ± 0.0	11.2 ± 0.0	14.2 ± 0.0
Fast0Tag (Zhang, Gong, and Shah, 2016)	21.6 ± 1.5	40.7 ± 0.6	28.7 ± 0.6	33.1 ± 0.7	30.7 ± 0.6	9.5 ± 0.0	23.4 ± 0.0	23.7 ± 0.1	13.6 ± 0.1	17.3 ± 0.1
Fast0Tag+	23.2 ± 0.3	42.1 ± 0.5	30.4 ± 0.3	35.1 ± 0.3	32.6 ± 0.3	10.0 ± 0.0	24.3 ± 0.2	24.6 ± 0.3	14.1 ± 0.2	17.9 ± 0.2
Ours (RankNet)	32.7 ± 0.4	54.0 ± 0.5	39.1 ± 0.5	45.1 ± 0.5	41.9 ± 0.5	10.2 ± 0.1	24.9 ± 0.4	25.5 ± 0.4	14.6 ± 0.3	18.5 ± 0.3
Ours (Hinge)	33.8 ± 0.2	55.0 ± 0.3	**39.7 ± 0.2**	**45.8 ± 0.2**	**42.5 ± 0.2**	10.5 ± 0.1	25.2 ± 0.1	25.5 ± 0.3	14.6 ± 0.2	18.6 ± 0.2
Ours (Fusion)	**34.1 ± 0.2**	**55.2 ± 0.2**	**39.7 ± 0.3**	**45.8 ± 0.4**	**42.5 ± 0.4**	**10.8 ± 0.1**	**25.8 ± 0.1**	**26.3 ± 0.2**	**15.0 ± 0.1**	**19.1 ± 0.1**

rank losses and their fusion are employed, respectively. Again, all the experiments are conducted with two different data split settings and evaluated under three evaluation scenarios, as described in Section 5.2. For reliability, we report the mean and the SEM of results ($k = 5$ used in evaluation metrics, i.e., Eqs. (19)–(21)) over three randomly generated known/unseen label splits under each evaluation scenario.

For the IFS setting, it is seen from Table 5 that all the models perform better than random guess in most of evaluation scenarios. However, DSP and ConSE result in the poorer performance than random guess in the unseen-action only scenario on Breakfast in terms of some specific metric, e.g., I-MAP. Overall, DSP and ConSE under-perform other methods considerably in terms of all five evaluation metrics under all three evaluation scenarios. Such results demonstrate that simply combining semantic representations of co-occurred multiple labels into one collective representation leads to catastrophic loss of semantic information, which is mainly responsible for the poor performance of DSP and ConSE in multi-label recognition. COSTA and GCN perform comparably consistently in varying scenarios which shows the limitation of GCN when a moderate number of labels are involved and no extra label relationship is available. Note that the results of COSTA and GCN for known actions (i.e. KnownA) are the same since both of them employ the same SVM classifiers for known actions. Fast0Tag generally outperforms COSTA and GCN on two datasets in terms of most of evaluation metrics. While COSTA learns a classifier for each label separately without considering a relationship among co-occurred labels, the consideration of such a relationship in Fast0Tag accounts for the better performance. By incorporating the semantic embedding learning into Fast0Tag, Fast0Tag+, our extension of Fast0Tag, constantly improves the performance of its original version in most circumstances on two datasets. Once again, this result lends us evidence to justify the necessity of semantic embedding learning used in our framework for zero-shot multi-label ZSL. In contrast, our model trained with either RankNet loss or the hinge rank loss generally outperforms all five models significantly in terms of five evaluation metrics under different evaluation scenarios on two datasets, as highlighted with bold-font in Table 5. By comparing our model to Fast0Tag+, we see three main differences between them as follows: *visual representations*, *network architectures* for the visual model and *loss* functions. Regarding visual representations, our model uses the segment-based visual features for an instance while Fast0Tag+employs an instance-level holistic visual representation. For network architectures, we employ an LSTM layer with recurrent connections to capture temporal coherence among segments of a video clip while Fast0Tag+simply uses a feed-forward network. As described in Section 3.2.2, we use an alternative loss function to that in Fast0Tag. Thus, those differences together leverage our performance gain over Fast0Tag+, which yet again lends us evidence to support our proposed framework. Finally, it is observed from Table 5 that in the IFS setting, the RankNet and the hinge rank losses perform differently on two datasets; the hinge rank loss generally outperforms the RankNet loss on both datasets with the exceptions of unseen action only scenarios on Breakfast. Nevertheless, the fusion of two models trained with different losses leads to the best performance in most circumstances as highlighted with bold-font in Table 5. Such results reveal that two losses behave quite differently and the diversity can be exploited via fusion, which provides useful information to develop more effective rank loss functions.

For the LFS setting, experimental results suggest that most of the models in question have similar behaviour to that in the IFS setting, as shown in Table 5. Once again, DSP and ConSE generally perform worse than other models and even under-perform random guess on Charades in the unseen-action only scenario. While COSTA and GCN yield better performance than DSP and ConSE overall, they generally under-perform Fast0Tag, Fast0Tag+and ours in all three evaluation scenarios. It is noteworthy that GCN performs no better than RGS for unseen actions in terms of all metrics except L-MAP. These results further validate the limitation of GCN when extra information of relations between labels is not available. In the LFS setting, our model trained with different rank losses generally outperforms others in most circumstances except for the unseen-action only scenario on Breakfast where Fast0Tag+performs better than ours marginally in terms of I-MAP. Regarding two rank losses used in our experiments, the hinge rank loss marginally outperforms the RankNet loss in most circumstances on two datasets. Once again, the fusion of results brought by two rank losses further improves the performance in most circumstances, which provides the further evidence on the complementary aspect of two different rank losses. As described in Section 5.2, the LFS setting is more challenging than the IFS setting and some salient visual features on test instances corresponding to unseen actions could completely miss in training examples. In this case, the use of a segment-level based visual representation and an LSTM layer in the visual model may not be able to generalize well due to a lack of training examples. Although such a result does not sufficiently favour the use of a segment-level based visual representation and an LSTM layer in the visual model in the presence of limited training data, it is no doubt that introducing a semantic model to Fast0Tag leverages the performance gain. Once again, experimental results here along with those compared to the baseline models under our LFS setting reveal a *training data sparsity* issue that has to be addressed in any future multi-label zero-shot human action recognition study.

Furthermore, Table 6 shows the experimental results in conventional multi-label human action recognition, i.e., all the actions are known in learning. In this circumstance, only the IFS setting is applicable. Hence, we use the same IFS setting as described in Section 4.1.2 but, unlike what has been done for simulating a zero-shot scenario, do not reserve any actions. Also we use the same procedure as done for zero-shot learning to search for optimal hyper-parameters for five models and ours and repeat the experiments on the same data split as the IFS setting for three trials with different parameter initialization. As a result, we report the mean and standard deviation (std) of three-trial

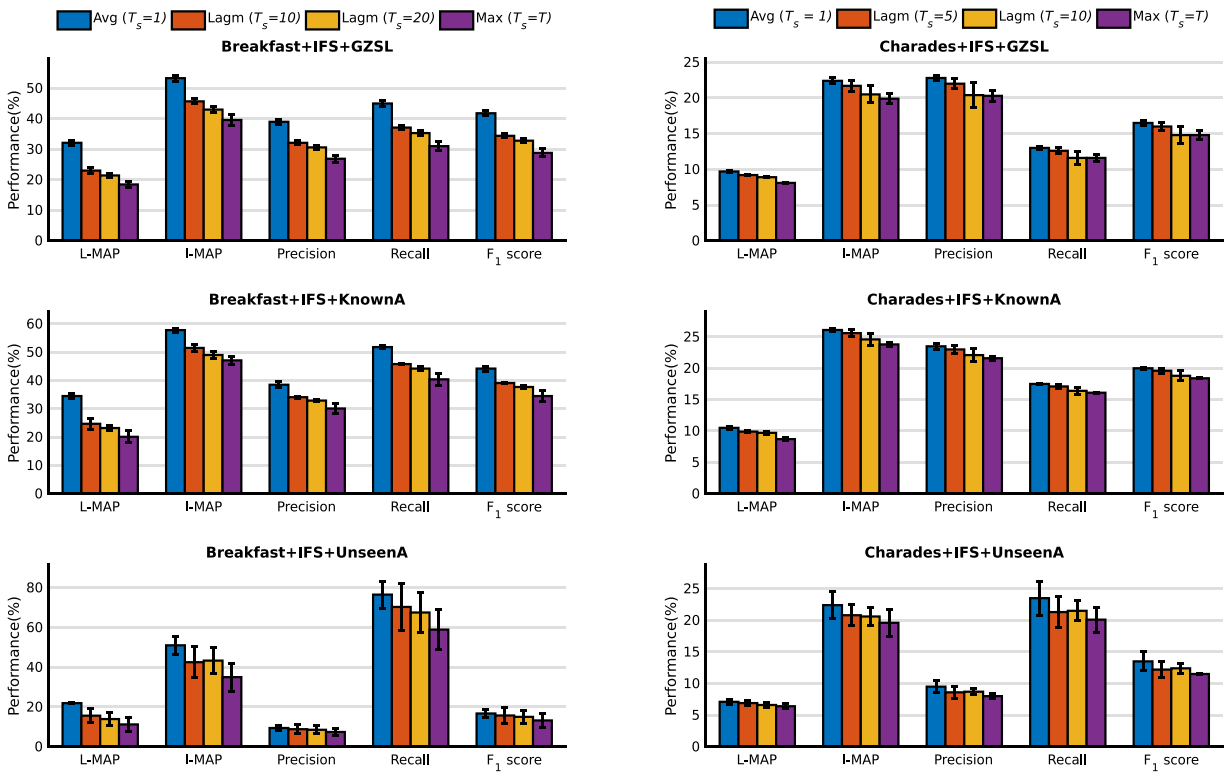

Fig. 4. Performance of different pooling strategies used to aggregate the temporal relatedness scores in the joint ranking embedding learning on two datasets, Breakfast and Charades, in three evaluation scenarios. Avg: average pooling; Lagm: local average global maximum pooling; Max: maximum pooling; T_s is the number of groups over which the pooling is done (c.f. Section 4.6.3).

results yielded by different methods. It is worth clarifying that we do not have any GCN results in Table 6 since GCN is not applicable for this experimental setting; in fact, all action label classifiers can be learned from training data and hence no additional GCN is required to predict unseen action classifiers. It is evident from Table 6 that our model trained with either of two rank losses as well as their fusion outperforms others in conventional multi-label recognition on both datasets. Without unseen classes, our model trained with the hinge rank loss generally performs slightly better than its counterpart trained with the RankNet loss. Once again, the fusion of results generated by those two models leads to the best performance. To see the degraded performance in a zero-shot scenario, we can compare the performance in the generalized ZSL evaluation scenario under the IFS setting, as shown in Table 5, to that reported in Table 6. By such a comparison, it is seen that the zero-shot performance of our model drops with a narrow margin (approximately less than 10% overall in terms of five different evaluation metrics). Given the fact that 10 out of 49 and 40 out of 157 human actions are reserved as unseen labels on Breakfast and Charades, respectively, this comparison on experimental results suggests that our proposed framework yields the promising performance for multi-label zero-shot human action recognition, which is close to the performance in multi-label human action recognition. Experimental results shown in Table 6 also suggest that other state-of-the-art methods behave similarly to ours in general. However, we also observe an unusual phenomenon from their performance; i.e., by a comparison to the generalized ZSL performance reported in Table 5, DSP yields slightly worse performance in multi-label recognition in terms of four of five evaluation metrics on Breakfast and so do Fast0Tag and Fast0Tag+in terms of L-MAP. By a closer look at the dataset and results in two experiments as well as our analysis, we find that at least two factors account for this unusual phenomenon: (a) co-occurred labels associated with most of video clips on Breakfast are redundant in light of semantics, and (b) the single collective semantic representation of co-occurred multiple labels used in DSP is insensitive to missing of few co-occurred labels due to the label information redundancy and the information loss resulting from the average operation in forming the single representation. Thus, we reckon that this phenomenon is rather specific to the nature of this dataset and the ZSL setting where there are only a small number of unseen labels.

5.4. Results on pooling strategy

We report the performance of three pooling strategies in terms of five evaluation criteria. It is evident from Fig. 4 the average pooling always performs the best and the maximum pooling performs the worst regardless of evaluation criteria. In addition, the local average global maximum pooling performs better when T_s is set smaller. Such results imply that our framework interprets the visual information at a global level that tends to recognize actions appearing in a video clip rather than a local level that identifies the accurate boundaries between different actions. From our empirical study, it is observed that the average pooling takes into account all information in a video to yield the relatedness scores while the maximum pooling uses only the local information regarding an abrupt change in visual domain but likely overlooks a large portion of useful information related to the nature of actions. Nevertheless, the maximum pooling might be beneficial for unsupervised action localization in the weak supervision setting, which is beyond the scope of this paper but worth studying in future.

In summary, our comparative study suggests that our proposed framework yields the favourable results and outperforms the existing state-of-the-art methods in general. The average pooling generally outperforms other alternatives in question.

DSP	ConSE	COSTA	GCN	Fast0Tag	Fast0Tag+	Ours (RankNet)	Ours (Hinge)	Ours (Fusion)
take_butter	**put_egg2plate**	**crack_egg**	**crack_egg**	**crack_egg**	smear_butter	**crack_egg**	**crack_egg**	**crack_egg**
stir_egg	**crack_egg**	**put_egg2plate**	**put_egg2plate**	**add_salt_pepper**	cut_bun	**put_egg2plate**	**butter_pan**	**put_egg2plate**
take_eggs	**butter_pan**	**butter_pan**	**butter_pan**	**put_egg2plate**	**put_egg2plate**	**fry_egg**	**take_plate**	**take_palte**
crack_egg	pour_egg2pan	cut_bun	cut_bun	pour_milk	**add_salt_pepper**	**add_salt_pepper**	**put_egg2plate**	**butter_pan**
fry_egg	**fry_egg**	**add_salt_pepper**	**add_salt_pepper**	fry_pancake	**crack_egg**	stir_fry_egg	**add_salt_pepper**	**add_salt_pepper**

Fig. 5. A test video clip in the IFS setting and the top-5 labels predicted by different methods (c.f. Section 4.6.2). Its ground-truth labels are **take_bowl**, **crack_egg**, **put_egg2plate**, **take_plate**, ***stir_egg***, **pour_egg2pan**, **stir_fry_egg**, **add_salt_pepper**, **butter_pan**.

DSP	ConSE	COSTA	GCN	Fast0Tag	Fast0Tag+	Ours (RankNet)	Ours (Hinge)	Ours (Fusion)
pour_milk	crack_egg	crack_egg	crack_egg	crack_egg	pour_milk	pour_milk	pour_milk	pour_milk
stir_egg	pour_egg2pan	put_egg2plate	put_egg2plate	pour_milk	stir_milk	add_teabag	spoon_powder	spoon_powder
pour_flour	fry_egg	pour_milk	pour_milk	fry_pancake	spoon_powder	spoon_powder	stir_milk	stir_milk
stir_milk	stir_fry_egg	add_salt_pepper	add_salt_pepper	put_egg2plate	add_teabag	pour_water	spoon_sugar	add_teabag
pour_sugar	put_egg2plate	fry_egg	fry_egg	take_plate	pour_water	stir_milk	pour_water	pour_water

Fig. 6. A test video clip in the IFS setting and the top-5 labels predicted by different methods (c.f. Section 4.6.2). Its ground-truth labels are **cut_orange**, **squeeze_orange**, **pour_juice**.

DSP	ConSE	COSTA	GCN	Fast0Tag	Fast0Tag+	Ours (RankNet)	Ours (Hinge)	Ours (Fusion)
take_butter	**put_egg2plate**	**crack_egg**	**crack_egg**	**crack_egg**	smear_butter	**crack_egg**	**crack_egg**	**crack_egg**
stir_egg	**crack_egg**	**put_egg2plate**	**put_egg2plate**	**add_salt_pepper**	cut_bun	**put_egg2plate**	**butter_pan**	**put_egg2plate**
take_eggs	**butter_pan**	**butter_pan**	**butter_pan**	**put_egg2plate**	**put_egg2plate**	**fry_egg**	**take_plate**	**take_palte**
crack_egg	pour_egg2pan	cut_bun	cut_bun	pour_milk	**add_salt_pepper**	**add_salt_pepper**	**put_egg2plate**	**butter_pan**
fry_egg	**fry_egg**	**add_salt_pepper**	**add_salt_pepper**	fry_pancake	**crack_egg**	stir_fry_egg	**add_salt_pepper**	**add_salt_pepper**

Fig. 7. A test video clip in the IFS setting and the top-5 labels predicted by different methods (c.f. Section 4.6.2). Its ground-truth labels are **crack_egg**, **fry_egg**, **put_egg2plate**, **take_plate**, **add_salt_pepper**, **butter_pan**.

Also, our experimental results demonstrate challenges in multi-label ZSL via our novel LFS setting especially when training data are less correlated to test instances associated with unseen classes in both semantic and visual domains.

5.5. Visual inspection

In general, visual inspection provides a manner that helps us understand the behaviour of a method intuitively. To gain an intuitive insight into the multi-label zero-shot human action recognition, we visualize a number of typical test video clips on Breakfast and the top-5 labels predicted by different state-of-the-art methods described in Section 4.6.2 and ours in terms of semantic relatedness scores. Our visual inspection mainly focuses on understanding of the behaviour of our model and issues arising from our work. As a result, Figs. 5–8 illustrate several key frames to human actions in typical test video clips and the top-5 predicted labels by different methods, where a correctly predicted known label is highlighted with bold font and a correctly predicted unseen label is marked with bold-italic font.

For the IFS setting, Figs. 5–7 illustrate three typical results yielded by different methods. Fig. 5 exemplifies the success of our model, where four out of the top-5 labels predicted by our model are the ground-truth actions and no other methods can match the performance of our model. This exemplified test instance suggests that the use of an LSTM layer in our visual model facilitates the recognition of distinctive actions in a video clip. Fig. 6 shows a test instance where all the methods fail to have any ground-truth labels in their top-5 predicted labels. Our visual inspection on this test instance reveals that non-trivial objects pertaining to different actions are concentrated in a small region located in top-right of frames in this video clip. Thus, it is extremely difficult to capture the useful information in the visual domain, which poses a challenge to all the existing human action recognition techniques. Figs. 5–8 reveal that our models trained with two rank losses yield different results for a test instance. Specially in Fig. 8, three of the top-5 labels predicted by two models are in common, however, the fusion method described in Section 3.3 successfully predicts five ground truth labels. These instances vividly demonstrate the different aspects of two rank losses and the synergy achieved by their fusion. Besides, these test instances illustrated in Figs. 5–8 also provide some insight regarding the behaviour of other state-of-the-art models used in our comparative study. For example, ConSE is more likely to yield the labels regarding

Data Split	DSP	ConSE	COSTA	GCN	Fast0Tag	Fast0Tag+	Ours(RankNet)	Ours(Hinge)	Ours (Fusion)
IFS	***stir_egg***	**pour_egg2pan**	**put_egg2plate**	**put_egg2plate**	**crack_egg**	**crack_egg**	**crack_egg**	**crack_egg**	**crack_egg**
	crack_egg	pour_milk	pour_milk	pour_milk	fry_pancake	pour_milk	**put_egg2plate**	**take_plate**	**put_egg2plate**
	take_eggs	pour_flour	**crack_egg**	**crack_egg**	**put_egg2plate**	***stir_egg***	**add_salt_pepper**	pour_oil	**take_plate**
	take_butter	pour_oil	pour_oil	pour_oil	pour_milk	**put_egg2plate**	***stir_egg***	**put_egg2plate**	**add_salt_pepper**
	pour_flour	pour_sugar	**add_salt_pepper**	**add_salt_pepper**	**add_salt_pepper**	pour_oil	fry_egg	**add_salt_pepper**	**Stir_fry_egg**
LFS	squeeze_orange	pour_sugar	pour_sugar	pour_sugar	pour_milk	pour_milk	pour_juice	squeeze_orange	squeeze_orange
	spoon_powder	pour_flour	**add_salt_pepper**	**add_salt_pepper**	spoon_powder	spoon_powder	squeeze_orange	cut_orange	cut_orange
	stir_egg	pour_milk	pour_milk	pour_milk	fry_egg	stir_milk	cut_orange	pour_juice	pour_juice
	stir_milk	pour_juice	spoon_powder	spoon_powder	stir_milk	cut_orange	take_galss	take_glass	**take_plate**
	cut_bun	pour_dough2pan	put_bun_together	put_bun_together	pour_juice	pour_juice	**take_plate**	**take_plate**	take_galss

Fig. 8. A test video clip appearing in the IFS and LFS settings and the top-5 labels predicted by different methods (c.f. Section 4.6.2) in two data split settings. Its ground truth labels are **take_bowl**, **crack_egg**, **put_egg2plate**, **take_plate**, ***stir_egg***, **pour_egg2pan**, **stir_fry_egg**, **add_salt_pepper**, **butter_pan** in the IFS setting, and ***take_bowl***, **crack_egg**, **put_egg2plate**, **take_plate**, ***stir_egg***, ***pour_egg2pan***, **stir_fry_egg**, **add_salt_pepper**, ***butter_pan*** in the LFS setting, respectively.

frequently used words in a human action domain. For those test instances shown in Figs. 5–7, at least four out of the top-5 labels predicted by ConSE are regarding different actions taken on "egg". For the instance shown in Fig. 8, all the top-5 labels predicted by ConSE are completely regarding "pour" actions commonly taken in kitchen. This limitation is due to the fact that ConSE uses a single collective semantic representation resulting from averaging the semantic representations of multiple co-occurred labels, which favours those frequently used word vectors but diminishes the opportunity of finding out infrequently used word vectors in prediction. We can also see that COSTA and GCN always predict the same top-5 results which are all known actions. These results suggest that the methods aiming to predict unseen action classifiers such as COSTA and GCN are more likely to rank known labels ahead of unseen ones.

Experimental results reported in Tables 3–5 suggest that all the models including ours generally perform worse under the LFS setting than under the IFS setting. On the one hand, the LFS setting results in a *training data sparsity* issue in contrast to the IFS setting. To see this issue, let us take the first split on Breakfast as an example. In this split shown in Table 2, there are 1196 and 1019 training examples in the IFS and the LFS setting, respectively. However, the number of training examples pertaining to specific known actions is significantly different in two split settings due to different data split protocols described in Section 4.1.2. For example, there are 330, 217, 254, 109 and 156 training examples with target labels,"crack_egg", "put_egg2plate", "take_plate", "stir_fry_egg" and "add_salt_pepper", respectively, in the IFS setting. In contrast, there are only 23, 23, 81, 23 and 20 examples with the same target labels, respectively, in the LFS setting. On the other hand, there is a major difference between those models and ours; i.e., our visual model employs a hidden layer of recurrent connection to capture temporal coherence underlying intrinsic visual features while those state-of-the-art models used in our comparative study do not have such a mechanism. It is well known that a learning model of a higher complexity or a larger capacity demands more informative training data. To this end, the training data sparsity issue affects the performance of our model more severely than other models; it is evident that the performance gain from the use of an LSTM layer in our visual model disappears due to a lack of sufficient training data required in training our visual model for capture temporal coherence.

To understand the difference between the IFS and the LFS settings and the training data sparsity issue intuitively, we illustrate the results yielded by the state-of-the-art methods and ours on a common instance appearing in test sets in two data split settings, as shown in Fig. 8. It is evident that four out of the top-5 action labels predicted by our model are the ground truth and all other models can predict some of ground-truth actions correctly under the IFS setting. In contrast, however, none of the models correctly predicts more than one ground-truth action for this exactly same test instance under the LFS setting. The visual inspection on this test instance clearly demonstrates the distinction between two data split settings; i.e., visual features associated with unseen actions are available in the IFS setting (an unrealistic scenario) but unavailable in the LFS setting (a realistic scenario), and the *training data sparsity* issue in the LFS setting, which poses a big challenge to all the existing multi-label ZSL methods including ours.

5.6. Model complexity

The architecture complexity of our learning model depends on the number of hidden layers, hidden units and their types as well as their connections used in neural networks to implement the visual and the semantic models for a dataset.

In our current implementation, the number of parameters in the visual model varies from 4.9 to 24.7 millions, and the number of parameters in the semantic model varies from 0.15 to 0.77 millions under different hyper-parameter settings. Obviously, the semantic model has much fewer parameters compared with the visual model, suggesting that introducing a semantic model does not incur a much higher computational burden but leads to the performance gain. In general, our model often takes longer training time than other state-of-the-art learning models used in our comparative study due to the use of a LSTM layer to capture temporal coherence.

Practically, with a GTX1080Ti GPU, the averaging time spent for training our learning model is roughly 13 min on Charades (i.e., 40 s per epoch multiplies approximately 20 epochs) and one hour on Breakfast due to a larger number of time steps ($T = 300$). One limitation of our learning model is a large memory requirement for training. Recall that the visual representations have to be reserved for use in the training of semantic model, it is required to load one large matrix with a size of $n \times d_e \times T$ into memory . In our implementation, the amount of GPU memory used for training on Charades and Breakfast is 3.5 GB and 10.5 GB, respectively.

6. Concluding remarks

In this paper, we have formulated human action recognition as a multi-label zero-shot learning problem and provide an effective solution by proposing a novel framework via joint latent ranking

Table 7
Optimal hyper-parameter values of different learning models found by grid search. **Notation**: IFS — Instance-First Split; LFS — Label-First Split; V — Visual model; S — Semantic model; lr — learning rate; C, ϵ — soft-margin and percentage of support vectors in SVM/SVR; m — margin in the hinge ranking loss. $N_1 \rightarrow N_2 \rightarrow d_e$ indicates a neural network architecture where N_1(dropout rate) is the number of neurons in the first hidden layer and the dropout rate used in learning; N_2 is the number of hidden neurons in the second hidden layer; and d_e is the number of neurons in the latent embedding layer.

Dataset	Data split	Model	Split 1	Split 2	Split 3
Breakfast	IFS	NRC (RankNet)	V : lr = $1e-4$; 1024(0.5) → 2048 → 500 S : lr = $1e-6$; 500 → 500	V : lr = $1e-4$; 1024(0.5) → 2048 → 500 S : lr = $1e-6$; 500 → 500	V : lr = $1e-4$; 2048(0.5) → 1024 → 500 S : lr = $1e-6$; 500 → 500
		NRC (Hinge)	V : lr = $1e-4$; 1024(0.5) → 2048 → 500 S : lr = $1e-6$; 500 → 500; $m = 1$	V : lr = $1e-4$; 1024(0.5) → 2048 → 500 S : lr = $1e-6$; 500 → 500; $m = 1$	V : lr = $1e-4$; 1024(0.5) → 2048 → 500 S : lr = $1e-6$; 700 → 500; $m = 1$
		WSE(RankNet)	V : lr = $1e-4$; 512(0) → 2048	V : lr = $1e-4$; 512(0) → 1024	V : lr = $1e-4$; 1024(0) → 2048
		WSE(Hinge)	V : lr = $1e-4$; 1024(0.5) → 2048; $m = 1$	V : lr = $1e-4$; 512(0) → 1024; $m = 1$	V : lr = $1e-4$; 1024(0) → 1024; $m = 1$
		RLR (RankNet)	V : lr = $1e-4$; 512(0) → 2048 → 800 S : lr = $1e-6$; 700 → 800	V : lr = $1e-4$; 256(0.5) → 1024 → 500 S : lr = $1e-6$; 500 → 500	V : lr = $1e-4$; 256(0.5) → 2048 → 200 S : lr = $1e-6$; 300 → 200
		RLR (Hinge)	V : lr = $1e-4$; 1024(0) → 2048 → 500 S : lr = $1e-6$; 700 → 500; $m = 10$	V : lr = $1e-4$; 512(0.5) → 1024 → 500 S : lr = $1e-6$; 500 → 500; $m = 10$	V : lr = $1e-4$; 512(0.5) → 2048 → 200 S : lr = $1e-6$; 500 → 200; $m = 10$
		DSP	$C = 1, \epsilon = 0.1$	$C = 1, \epsilon = 0.1$	$C = 1, \epsilon = 0.1$
		ConSE	$C = 1$	$C = 1$	$C = 1$
		COSTA	$C = 1$	$C = 1$	$C = 1$
		Fast0Tag	V : lr = $1e-4$; 8192(0) → 1024	V : lr = $1e-4$; 8192(0.5) → 2048	V : lr = $1e-2$; 8192(0.5) → 2048
		Fast0Tag+	V : lr = $1e-4$; 4096(0) → 1024 → 800 S : lr = $1e-6$; 500 → 800	V : lr = $1e-4$; 8192(0.5) → 2048 → 800 S : lr = $1e-6$; 700 → 800	V : lr = $1e-4$; 8192(0) → 1024 → 200 S : lr = $1e-6$; 300 → 200
		Ours (RankNet)	V : lr = $1e-4$; 512(0) → 2048 → 800 S : lr = $1e-6$; 700 → 800	V : lr = $1e-4$; 1024(0) → 2048 → 500 S : lr = $1e-6$; 700 → 500	V : lr = $1e-4$; 1024(0) → 2048 → 500 S : lr = $1e-6$; 700 → 500
		Ours (Hinge)	V : lr = $1e-4$; 256(0.5) → 2048 → 500 S : lr = $1e-6$; 700 → 500; $m = 1$	V : lr = $1e-4$; 256(0.5) → 1024 → 500 S : lr = $1e-6$; 700 → 500; $m = 1$	V : lr = $1e-4$; 1024(0) → 1024 → 500 S : lr = $1e-6$; 500 → 500; $m = 1$
	LFS	NRC (RankNet)	V : lr = $1e-4$; 2048(0.5) → 2048 → 200 S : lr = $1e-6$; 500 → 200	V : lr = $1e-4$; 2048(0) → 2048 → 800 S : lr = $1e-6$; 500 → 800	V : lr = $1e-4$; 2048(0) → 2048 → 500 S : lr = $1e-6$; 700 → 500
		NRC (Hinge)	V : lr = $1e-4$; 4096(0) → 2048 → 500 S : lr = $1e-6$; 300 → 500; $m = 10$	V : lr = $1e-4$; 4096(0) → 2048 → 500 S : lr = $1e-6$; 300 → 500; $m = 10$	V : lr = $1e-4$; 2048(0.5) → 2048 → 200 S : lr = $1e-6$; 500 → 200; $m = 1$
		WSE(RankNet)	V : lr = $1e-4$; 1024(0) → 1024	V : lr = $1e-4$; 1024(0) → 1024	V : lr = $1e-4$; 1024(0.5) → 1024
		WSE(Hinge)	V : lr = $1e-4$; 1024(0.5) → 2048; $m = 10$	V : lr = $1e-4$; 1024(0.5) → 2048; $m = 10$	V : lr = $1e-4$; 512(0) → 2048; $m = 1$
		RLR (RankNet)	V : lr = $1e-4$; 256(0.5) → 1024 → 500 S : lr = $1e-6$; 500 → 500	V : lr = $1e-4$; 512(0.5) → 1024 → 200 S : lr = $1e-6$; 700 → 200	V : lr = $1e-4$; 512(0.5) → 2048 → 200 S : lr = $1e-6$; 500 → 200
		RLR (Hinge)	V : lr = $1e-4$; 512(0) → 2048 → 800 S : lr = $1e-6$; 500 → 800; $m = 10$	V : lr = $1e-4$; 256(0) → 1024 → 800 S : lr = $1e-6$; 700 → 800; $m = 10$	V : lr = $1e-4$; 512(0) → 2048 → 800 S : lr = $1e-6$; 700 → 800; $m = 10$
		DSP	$C = 100, \epsilon = 0.1$	$C = 100, \epsilon = 0.1$	$C = 100, \epsilon = 0.1$
		ConSE	$C = 100$	$C = 100$	$C = 100$
		COSTA	$C = 100$	$C = 100$	$C = 100$
		Fast0Tag	V : lr = $1e-4$; 4096(0.5) → 2048	V : lr = $1e-4$; 4096(0) → 1024	V : lr = $1e-2$; 8192(0.5) → 2048
		Fast0Tag+	V : lr = $1e-4$; 8192(0.5) → 1024 → 800 S : lr = $1e-6$; 500 → 800	V : lr = $1e-4$; 8192(0.5) → 1024 → 800 S : lr = $1e-6$; 500 → 800	V : lr = $1e-4$; 4096(0) → 1024 → 200 S : lr = $1e-6$; 300 → 200
		Ours (RankNet)	V : lr = $1e-4$; 512(0) → 1024 → 800 S : lr = $1e-6$; 500 → 800	V : lr = $1e-4$; 256(0) → 1024 → 800 S : lr = $1e-6$; 700 → 800	V : lr = $1e-4$; 512(0.5) → 1024 → 200 S : lr = $1e-6$; 700 → 200
		Ours (Hinge)	V : lr = $1e-4$; 512(0.5) → 1024 → 200 S : lr = $1e-6$; 500 → 200; $m = 10$	V : lr = $1e-4$; 512(0.5) → 1024 → 800 S : lr = $1e-6$; 500 → 800; $m = 10$	V : lr = $1e-4$; 256(0) → 2048 → 500 S : lr = $1e-6$; 500 → 500; $m = 1$
Charades	IFS	NRC (RankNet)	V : lr = $1e-4$; 1024(0) → 1024 → 500 S : lr = $1e-6$; 500 → 500	V : lr = $1e-4$; 1024(0) → 1024 → 500 S : lr = $1e-6$; 500 → 500	V : lr = $1e-4$; 2048(0) → 1024 → 200 S : lr = $1e-6$; 700 → 200
		NRC (Hinge)	V : lr = $1e-4$; 2048(0) → 2048 → 800 S : lr = $1e-6$; 700 → 800; $m = 10$	V : lr = $1e-4$; 2048(0) → 2048 → 800 S : lr = $1e-6$; 500 → 800; $m = 10$	V : lr = $1e-4$; 4096(0.5) → 2048 → 500 S : lr = $1e-6$; 300 → 500; $m = 10$
		WSE(RankNet)	V : lr = $1e-4$; 1024(0.5) → 2048	V : lr = $1e-4$; 512(0.5) → 2048	V : lr = $1e-4$; 256(0.5) → 2048
		WSE(Hinge)	V : lr = $1e-4$; 1024(0.0) → 2048	V : lr = $1e-4$; 1024(0.0) → 1024	V : lr = $1e-4$; 1024(0.0) → 2048
		RLR (RankNet)	V : lr = $1e-4$; 256(0.5) → 1024 → 500 S : lr = $1e-6$; 500 → 500	V : lr = $1e-4$; 256(0.5) → 1024 → 500 S : lr = $1e-6$; 500 → 500	V : lr = $1e-4$; 256(0.5) → 1024 → 500 S : lr = $1e-6$; 500 → 500
		RLR (Hinge)	V : lr = $1e-4$; 1024(0.5) → 2048 → 500 S : lr = $1e-6$; 700 → 500; $m = 10$	V : lr = $1e-4$; 1024(0.5) → 2048 → 500 S : lr = $1e-6$; 700 → 500; $m = 10$	V : lr = $1e-4$; 1024(0.5) → 2048 → 500 S : lr = $1e-6$; 700 → 500; $m = 10$
		DSP	$C = 1, \epsilon = 0.1$	$C = 1, \epsilon = 0.1$	$C = 1, \epsilon = 0.1$
		ConSE	$C = 1$	$C = 1$	$C = 1$
		COSTA	$C = 1$	$C = 1$	$C = 1$
		Fast0Tag	V : lr = $1e-4$; 4096(0.5) → 2048	V : lr = $1e-4$; 8192(0) → 2048	V : lr = $1e-4$; 4096(0.5) → 2048
		Fast0Tag+	V : lr = $1e-4$; 8192(0.5) → 2048 → 800 S : lr = $1e-6$; 700 → 800	V : lr = $1e-4$; 8192(0.5) → 2048 → 800 S : lr = $1e-6$; 700 → 800	V : lr = $1e-4$; 8192(0.5) → 1024 → 800 S : lr = $1e-6$; 500 → 800
		Ours (RankNet)	V : lr = $1e-4$; 256(0.5) → 2048 → 800 S : lr = $1e-6$; 700 → 800	V : lr = $1e-4$; 256(0.5) → 2048 → 800 S : lr = $1e-6$; 700 → 800	V : lr = $1e-4$; 1024(0.5) → 1024 → 800 S : lr = $1e-6$; 500 → 800
		Ours (Hinge)	V : lr = $1e-4$; 256(0.5) → 2048 → 500 S : lr = $1e-6$; 700 → 500; $m = 1$	V : lr = $1e-4$; 256(0.5) → 2048 → 800 S : lr = $1e-6$; 700 → 800; $m = 1$	V : lr = $1e-4$; 1024(0.5) → 1024 → 800 S : lr = $1e-6$; 500 → 800; $m = 1$
	LFS	NRC (RankNet)	V : lr = $1e-4$; 1024(0) → 2048 → 800 S : lr = $1e-6$; 700 → 800	V : lr = $1e-4$; 2048(0) → 1024 → 200 S : lr = $1e-6$; 700 → 200	V : lr = $1e-4$; 4096(0) → 2048 → 500 S : lr = $1e-6$; 300 → 500
		NRC (Hinge)	V : lr = $1e-4$; 2048(0) → 2048 → 500 S : lr = $1e-6$; 500 → 800; $m = 10$	V : lr = $1e-4$; 2048(0) → 1024 → 800 S : lr = $1e-6$; 700 → 200; $m = 1$	V : lr = $1e-4$; 2048(0) → 2048 → 500 S : lr = $1e-6$; 500 → 800; $m = 10$
		WSE(RankNet)	V : lr = $1e-4$; 1024(0.5) → 2048	V : lr = $1e-4$; 512(0.5) → 2048	V : lr = $1e-4$; 1024(0.5) → 2048
		WSE(Hinge)	V : lr = $1e-4$; 1024(0.0) → 2048; $m = 10$	V : lr = $1e-4$; 256(0.0) → 2048; $m = 1$	V : lr = $1e-4$; 1024(0.5) → 2048; $m = 10$
		RLR (RankNet)	V : lr = $1e-4$; 256(0.5) → 1024 → 500 S : lr = $1e-6$; 500 → 500	V : lr = $1e-4$; 256(0.5) → 2048 → 800 S : lr = $1e-6$; 700 → 800	V : lr = $1e-4$; 256(0.5) → 2048 → 800 S : lr = $1e-6$; 700 → 800
		RLR (Hinge)	V : lr = $1e-4$; 512(0.5) → 2048 → 800 S : lr = $1e-6$; 700 → 800; $m = 10$	V : lr = $1e-4$; 512(0.5) → 2048 → 800 S : lr = $1e-6$; 700 → 800; $m = 10$	V : lr = $1e-4$; 512(0.5) → 2048 → 800 S : lr = $1e-6$; 500 → 800; $m = 10$
		DSP	$C = 1, \epsilon = 0.1$	$C = 1, \epsilon = 0.1$	$C = 1, \epsilon = 0.1$
		ConSE	$C = 1$	$C = 1$	$C = 1$
		COSTA	$C = 1$	$C = 1$	$C = 1$
		Fast0Tag	V : lr = $1e-4$; 4096(0.5) → 1024	V : lr = $1e-4$; 8192(0.5) → 2048	V : lr = $1e-4$; 4096(0.5) → 1024
		Fast0Tag+	V : lr = $1e-4$; 8192(0.5) → 2048 → 800 S : lr = $1e-6$; 700 → 800	V : lr = $1e-4$; 8192(0) → 1024 → 500 S : lr = $1e-6$; 500 → 500	V : lr = $1e-4$; 4096(0.5) → 1024 → 500 S : lr = $1e-6$; 500 → 500
		Ours (RankNet)	V : lr = $1e-4$; 512(0.5) → 2048 → 800 S : lr = $1e-6$; 700 → 800	V : lr = $1e-4$; 512(0.5) → 2048 → 800 S : lr = $1e-6$; 500 → 800	V : lr = $1e-4$; 1024(0.5) → 2048 → 500 S : lr = $1e-6$; 700 → 500
		Ours (Hinge)	V : lr = $1e-4$; 512(0.5) → 1024 → 500 S : lr = $1e-6$; 700 → 500; $m = 1$	V : lr = $1e-4$; 256(0.5) → 2048 → 500 S : lr = $1e-6$; 700 → 500; $m = 1$	V : lr = $1e-4$; 1024(0.5) → 1024 → 800 S : lr = $1e-6$; 500 → 800; $m = 1$

embedding learning. To carry out our framework, we employ a neural network of the heterogeneous architecture for visual embedding, where an LSTM layer is used to facilitate capturing temporal coherence information underlying different actions from weakly annotated video data. Also, we advocate the use of semantic embedding learning to facilitate bridging the semantic gap and effective knowledge transfer, which is implemented by a feed-forward neural network. All the above contributions have been thoroughly verified via our comparative study with various well-motivated settings. Experimental results on two benchmark multi-label human action datasets suggest that our proposed framework generally outperforms not only the baseline systems but also several state-of-the-art multi-label ZSL approaches in all the different test scenarios.

Although we have demonstrated favourable results on two benchmark datasets in comparison to state-of-the-art approaches, our observations on the performance of all the approaches used in our comparative study including ours suggest that the existing multi-label ZSL techniques are not ready for a real application; the instance-first split setting fails to simulate real multi-label zero-shot human action recognition scenarios while the performance becomes even worse under the label-first split setting

that simulates a real scenario. Nevertheless, our experimental results including visual inspection provide the insightful information for improving our proposed framework. In our ongoing work, we would address issues arising from our experiments and observations with proper techniques. To address the *training data sparsity* issue revealed in our experiments, we would develop unsupervised learning algorithms to discover salient yet intrinsic visual features from unlabelled video clips and further incorporate proper temporal constraints into our rank loss functions to better capture temporal coherence. Also, the GAN-based synthetic feature generation idea could be further developed for weakly-supervised multi-label video clips to addressing the training data sparsity issue. Moreover, we would consider diverse pooling strategies and introduce attention mechanisms to our model for improving implicit salient feature extraction and accurate localization of different yet complex actions involved in a video clip during the visual embedding learning. Also, we would employ alternative semantic representations developed by ourselves (Wang & Chen, 2017a), which encode the semantic relatedness between action labels more accurately, in the semantic embedding learning to facilitate knowledge transfer.

While our framework is proposed especially for multi-label zero-shot human action recognition, we would highlight that it is directly applicable to multi-label human action recognition without modification as demonstrated in our experiments. Also, our framework is easy to adapt for tackling various multi-label ZSL problems in different domains. For example, we can apply our framework to miscellaneous multi-label zero-shot classification tasks on temporal or sequential data, e.g., acoustic event classification, straightforward as well as multi-label zero-shot learning tasks on static data, e.g., object recognition, by replacing a neural network of the heterogeneous architecture only with a neural network of only feed-forward connections in the visual model. Thus, we are going to explore such extensions and applications in our future work.

Acknowledgements

The authors would like to thank the anonymous reviewers for their comments that improve the presentation of this manuscript.

Appendix. Hyper-parameters

In this appendix, we report all the optimal hyper-parameter values of different learning models used in our experiments to enable one to replicate our experimental results. Table 7 summarizes all the optimal hyper-parameter values obtained with a grid-based search via cross-validation as described in Section 5.1.

References

Abu-El-Haija, S., Kothari, N., Lee, J., Natsev, P., Toderici, G., Varadarajan, B., et al. (2016). Youtube-8M: A large-scale video classification benchmark. arXiv preprint arXiv:1609.08675.

Akata, Z., Reed, S., Walter, D., Lee, H., & Schiele, B. (2015). Evaluation of output embeddings for fine-grained image classification. In *IEEE conference on computer vision and pattern recognition* (pp. 2927–2936).

Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., et al. (2005). Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on machine learning* (pp. 89–96). ACM.

Cabral, R., De la Torre, F., Costeira, J. P., & Bernardino, A. (2015). Matrix completion for weakly-supervised multi-label image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *37*(1), 121–135.

Changpinyo, S., Chao, W. -L., Gong, B., & Sha, F. (2016). Synthesized classifiers for zero-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5327–5336).

Changpinyo, S., Chao, W. -L., & Sha, F. (2017). Predicting visual exemplars of unseen classes for zero-shot learning. In *Proceedings of the IEEE international conference on computer vision* (pp. 3476–3485).

Chollet, F. (2015). *Keras, GitHub repository, 5bcac37*. GitHub, https://github.com/fchollet/keras.

Deng, J., Dong, W., Socher, R., Li, L. -J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition* (pp. 248–255). IEEE.

Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., et al. (2015). Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2625–2634).

Frome, A., Corrado, G. S., Shlens, J., Bengio, S., Dean, J., Mikolov, T., et al. (2013). Devise: A deep visual-semantic embedding model. *Advances in Neural Information Processing Systems*, 2121–2129.

Fu, Y., & Huang, T. (2010). Manifold and subspace learning for pattern recognition. *Pattern Recognition and Machine Vision*, *6*, 215.

Fu, Y., Yang, Y., Hospedales, T., Xiang, T., & Gong, S. (2014). Transductive multi-label zero-shot learning. In *Proceedings of British machine vision conference*.

Gong, Y., Jia, Y., Leung, T., Toshev, A., & Ioffe, S. (2013). Deep convolutional ranking for multilabel image annotation. arXiv preprint arXiv:1312.4894.

Gong, Y., Ke, Q., Isard, M., & Lazebnik, S. (2014). A multi-view embedding space for modeling internet images, tags, and their semantics. *International Journal of Computer Vision*, *106*(2), 210–233.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., et al. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems*, 2672–2680.

Gu, F., Sridhar, M., Cohn, A., Hogg, D., Flórez-Revuelta, F., Monekosso, D., et al. (2016). Weakly supervised activity analysis with spatio-temporal localisation. *Neurocomputing*, *216*, 778–789.

Guillaumin, M., Mensink, T., Verbeek, J., & Schmid, C. (2009). Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *IEEE international conference on computer vision* (pp. 309–316). IEEE.

Herbrich, R., Obermayer, K., & Graepel, T. (1999). Large margin rank boundaries for ordinal regression. In *Advances in large margin classifiers* (pp. 115–132). MIT Press, Cambridge MA.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Huang, H., Wang, C., Yu, P. S., & Wang, C. -D. (2019). Generative dual adversarial network for generalized zero-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 801–810).

Jiang, Y. -G., Wu, Z., Wang, J., Xue, X., & Chang, S. -F. (2017). Exploiting feature and class relationships in video categorization with regularized deep neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*(2), 352–364.

Karpathy, A., & Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *IEEE conference on computer vision and pattern recognition* (pp. 3128–3137).

Kavukcuoglu, K., Ranzato, M., & LeCun, Y. (2010). Fast inference in sparse coding algorithms with applications to object recognition. arXiv preprint arXiv:1010.3467.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. In *Proceedings of the 3rd international conference on learning representations*.

Kuehne, H., Arslan, A., & Serre, T. (2014). The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 780–787).

Lampert, C. H., Nickisch, H., & Harmeling, S. (2014). Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(3), 453–465.

Lapin, M., Hein, M., & Schiele, B. (2017). Analysis and optimization of loss functions for multiclass, top-k, and multilabel classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*(7), 1533–1554.

Lee, C. -W., Fang, W., Yeh, C. -K., & Frank Wang, Y. -C. (2018). Multi-label zero-shot learning with structured knowledge graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1576–1585).

Lei Ba, J., Swersky, K., Fidler, S., et al. (2015). Predicting deep zero-shot convolutional neural networks using textual descriptions. In *Proceedings of the IEEE international conference on computer vision* (pp. 4247–4255).

Li, X., Uricchio, T., Ballan, L., Bertini, M., Snoek, C. G., & Bimbo, A. D. (2016). Socializing the semantic gap: A comparative survey on image tag assignment, refinement, and retrieval. *ACM Computing Surveys*, *49*(1), 14.

Ma, C. -Y., Chen, M. -H., Kira, Z., & AlRegib, G. (2019). TS-LSTM and temporal-inception: Exploiting spatiotemporal dynamics for activity recognition. *Signal Processing: Image Communication*, *71*, 76–87.

Manning, C. D., Raghavan, P., & Schütze, H. (2009). An introduction to inf. retrieval. In *An introduction to inf. retrieval*. Cambridge University Press.

Mensink, T., Gavves, E., & Snoek, C. G. (2014). Costa: Co-occurrence statistics for zero-shot classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2441–2448).

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 3111–3119.

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th international conference on machine learning* (pp. 807–814).

Nam, J., Kim, J., Mencía, E. L., Gurevych, I., & Fürnkranz, J. (2014). Large-scale multi-label text classification – revisiting neural networks. In *Joint European conference on machine learning and knowledge discovery in databases* (pp. 437–452). Springer.

Nam, J., Mencía, E. L., Kim, H. J., & Fürnkranz, J. (2015). Predicting unseen labels using label hierarchies in large-scale multi-label learning. In *Joint European conference on machine learning and knowledge discovery in databases* (pp. 102–118). Springer.

Norouzi, M., Mikolov, T., Bengio, S., Singer, Y., Shlens, J., Frome, A., et al. (2014). Zero-shot learning by convex combination of semantic embeddings. In *International conference on learning representations*.

Ren, Z., Jin, H., Lin, Z., Fang, C., & Yuille, A. L. (2017). Multiple instance visual-semantic embedding. In *BMVC*.

Sandouk, U., & Chen, K. (2016a). Learning contextualized semantics from co-occurring terms via a siamese architecture. *Neural Networks*, *76*, 65–96.

Sandouk, U., & Chen, K. (2016b). Multi-label zero-shot learning via concept embedding. arXiv preprint arXiv:1606.00282.

Sigurdsson, G. A., Varol, G., Wang, X., Farhadi, A., Laptev, I., & Gupta, A. (2016). Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European conference on computer vision, ECCV* (pp. 510–526). Springer.

Sorower, M. S. (2010). *A literature survey on algorithms for multi-label learning*: *Technical Report*, Corvallis, U.S.A.: Oregon State University.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research (JMLR)*, *15*(1), 1929–1958.

Tang, P., Wang, X., Huang, Z., Bai, X., & Liu, W. (2017). Deep patch learning for weakly supervised object classification and discovery. *Pattern Recognition*, *71*(3), 446–459.

Tran, D., Bourdev, L., Fergus, R., Torresani, L., & Paluri, M. (2015). Learning spatiotemporal features with 3D convolutional networks. In *IEEE international conference on computer vision, ICCV* (pp. 4489–4497). IEEE.

Wang, Q., & Chen, K. (2017). Alternative semantic representations for zero-shot human action recognition. In *Proceedings of European conference on machine learning and principles and practice of knowledge discovery* (pp. 87–102).

Wang, Q., & Chen, K. (2017b). Zero-shot visual recognition via bidirectional latent embedding. *International Journal of Computer Vision*, *124*(3), 356—383.

Wang, Q., Jia, N., & Breckon, T. (2019). A baseline for multi-label image classification using ensemble deep CNN. In *IEEE international conference on image processing*.

Wang, H., & Schmid, C. (2013). Action recognition with improved trajectories. In *Proceedings of the IEEE international conference on computer vision* (pp. 3551–3558).

Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., & Xu, W. (2016). CNN-RNN: A unified framework for multi-label image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2285–2294).

Wang, X., Ye, Y., & Gupta, A. (2018). Zero-shot recognition via semantic embeddings and knowledge graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 6857–6866).

Wang, D., Yuan, Y., & Wang, Q. (2019). Early action prediction with generative adversarial networks. *IEEE Access*, *7*, 35795–35804.

Wei, Y., Xia, W., Lin, M., Huang, J., Ni, B., Dong, J., et al. (2016). HCP: A flexible CNN framework for multi-label image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*(9), 1901–1907.

Weston, J., Bengio, S., & Usunier, N. (2010). Large scale image annotation: Learning to rank with joint word-image embeddings. *Machine Learning*, *81*(1), 21–35.

Xian, Y., Akata, Z., Sharma, G., Nguyen, Q., Hein, M., & Schiele, B. (2016). Latent embeddings for zero-shot classification. In *IEEE conference on computer vision and pattern recognition* (pp. 69–77).

Xian, Y., Lorenz, T., Schiele, B., & Akata, Z. (2018). Feature generating networks for zero-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5542–5551).

Xian, Y., Schiele, B., & Akata, Z. (2017). Zero-shot learning - the good, the bad and the ugly. In *IEEE conference on computer vision and pattern recognition* (pp. 4582–4591).

Yu, Y., Ji, Z., Guo, J., & Pang, Y. (2018). Transductive zero-shot learning with adaptive structural embedding. *IEEE Transactions on Neural Networks and Learning Systems*, *29*(9), 4116–4127.

Yu, Y., Ji, Z., Guo, J., & Zhang, Z. (2018). Zero-shot learning via latent space encoding. *IEEE Transactions on Cybernetics*, (99), 1–12.

Yue-Hei Ng, J., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., & Toderici, G. (2015). Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4694–4702).

Zhang, Y., Acharyya, R., Liu, J., & Gong, B. (2016). Infinite-label learning with semantic output codes. arXiv preprint arXiv:1608.06608.

Zhang, Y., Gong, B., & Shah, M. (2016). Fast zero-shot image tagging. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5985–5994). IEEE.

Zhang, Z., & Saligrama, V. (2015). Zero-shot learning via semantic similarity embedding. In *IEEE international conference on computer vision* (pp. 4166–4174).

Zhang, Z., & Saligrama, V. (2016). Zero-shot learning via joint latent similarity embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 6034–6042).

Zhang, L., Xiang, T., & Gong, S. (2017). Learning a deep embedding model for zero-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2021–2030).

Zhang, M. L., & Zhou, Z. H. (2014). A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, *26*(8), 1819–1837.

Zhou, Z. -H., & Zhang, M. -L. (2007). Multi-instance multi-label learning with application to scene classification. *Advances in Neural Information Processing Systems*, *19*, 1609.