Matching Video Net: Memory-based embedding for video action recognition

Daesik Kim, Myunggi Lee, Nojun Kwak
Graduate School of Convergence Science and Technology
Seoul National University, Seoul, Korea
Email: daesik.kim@snu.ac.kr, myunggi89@snu.ac.kr, nojunk@snu.ac.kr

***Abstract*—Most of recent successful researches on action recognition are based on deep learning structures. Nonetheless, training deep neural networks is notorious for requiring huge amount of data. On the other hand, not enough data can lead to an overfitted model. In this work, we propose a novel model, *matching video net* (MVN), which can be trained with a small amount of data. In order to avoid the problem of overfitting, we use a non-parametric setup on top of parametric networks with external memories. An input clip of video is transformed into an embedding space and matched to the memorized samples in the embedding space. Then, the similarities between the input and the memorized data are measured to determine the nearest neighbors. We perform experiments in a supervised manner on action recognition datasets, achieving state-of-the-art results. Moreover, we applied our model to one-shot learning problems with a novel training strategy. Our model achieves surprisingly good results in predicting unseen action classes from only a few examples.**

I. Introduction

With huge advances of deep learning, large scale datasets are essentially required for training deep structures sufficiently. Optimizing enormous amount of parameters in a deep structure necessitates hundreds of thousands examples. Moreover, training a deep network is too slow because of many weight updates using stochastic gradient decent.

In particular, researches on video action recognition have established deep models that have millions of parameters to be trained. In order to optimize whole parameters in those models, a big dataset which includes over thousands of videos is needed. For instance, even though 3D convolution model,'C3D [1]' showed a state-of-the-art performance of video action recognition with UCF101 dataset [2], a full network of C3D needs Sports-1M dataset [3] which includes over 1 million videos to be optimized. Note that Sports-1M has 5 times the number of categories and 100 times the number of videos compared with UCF101.

However, humans learn new concepts with few examples. In contrast to recent deep learning systems, a child can generalize simple actions (e.g., walking) from few samples in real life. This motivates the interesting learning concept of "one-shot learning" which can learn a new concept from a single example. This idea of "one-shot learning" is also highly related to newly expanding a concept. While new target classes can be added in recent classification models, we have to retrain classifiers with whole data from the scratch. This can cause wastes of time and other computational resources. In case of one-shot learning, those troublesome can be efficiently overcome.

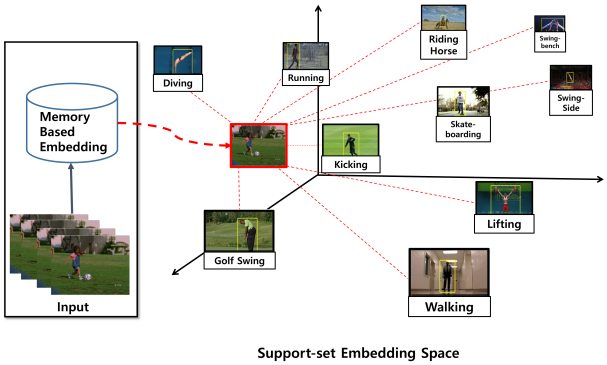

Fig. 1. Abstract concept of our proposed model based on distance metric. The memory based embedding network converts input sequences to a vector in a support-set embedding space. Hence, we can predict the class of the input by comparing distances between the input vector and the support-set.

Our work is motivated by the difficulties in training deep models with small dataset and in adding new target classes. Therefore, we propose a novel model combined with non-parametric method. The non-parametric model has several characteristics in training process. Some models in non-parametric methods (e.g., nearest neighbors) even do not require training process but the result depends on the chosen metric. In this work, we tried to adapt a non-parametric setup to a parametric model to resolve the aforementioned difficulties. Figure 1 briefly shows how a non-parametric method can be adapted to the action recognition problem.

Moreover, non-parametric methods can be developed with memory augmented network [4]. For example, when we recognize a human action from a scene in real life, we consider important scenes in our memory to compare with. Therefore we can use the external memory architecture as a long-term memory to store sampled representations.

In this paper, we make the following contributions:

- We propose a novel architecture which can efficiently recognize human actions with the non-parametric classification method from a small amount of data.
- We test our model on one-shot learning problems to predict unseen classes from a few examples without the fine-tuning process.

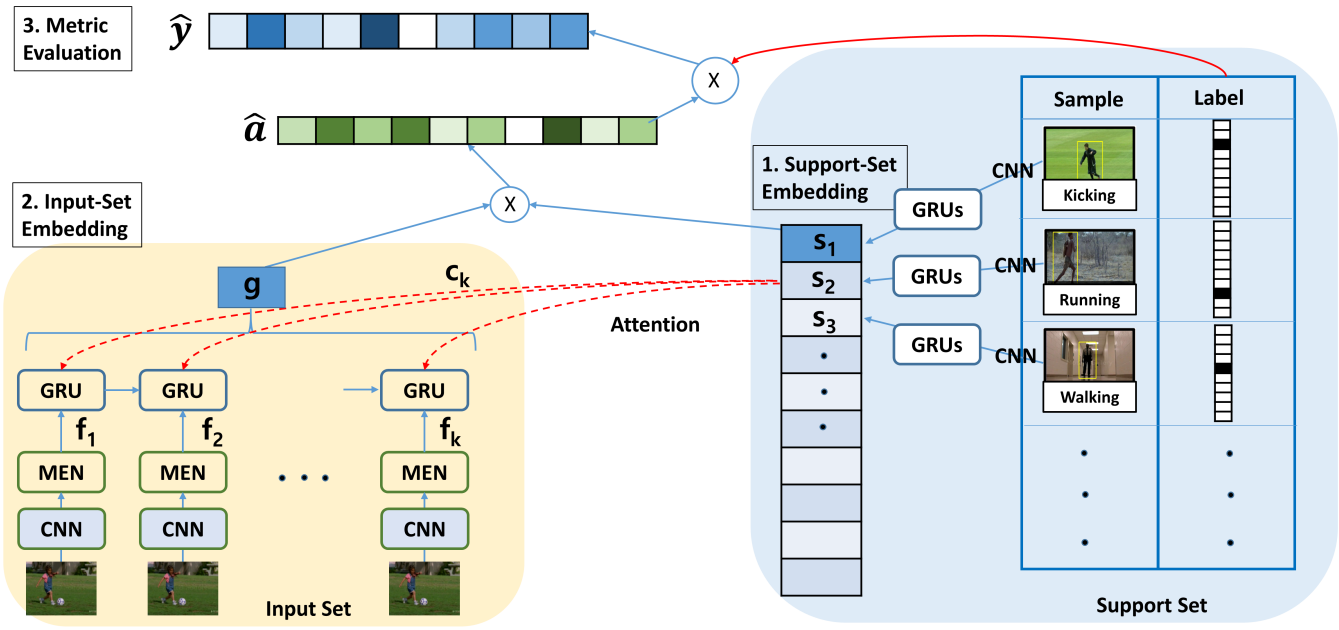

Fig. 2. Overall structure of our model. 1) Support set embedding S is acquired using GRU networks. 2) Input sequences are fed into MEN unit and then passed through a GRU network conditioned by support set embedding S. 3) A similarity vector $\hat{a}$ is obtained by calculating distances between the vector g and the matrix $S = \{(s_i, l_i)\}_{i=1}^{m}$. Lastly, a class prediction distribution $\hat{y}$ is obtained by multiplying the similarity vector $\hat{a}$ and one-hot vectors of support set's labels.

Our paper is organized as follows. Section 2 reviews the related work on action recognition and one-shot learning. In Section 3, we will introduce our proposed model including memory based embedding and metric based classification. In section 4, our experimental setting and three action recognition datasets are described. Then we demonstrate our results on supervised and one-shot learning. Finally, section 5 concludes our paper.

II. Related Work

1) Deep learning for action recognition: Most of recent works on action recognition have made remarkable improvements in the benefit of successful deep architectures. Over the years, convolutional neural networks (CNN) and recurrent neural networks (RNN) have been applied in several ways to produce state-of-the-art results on action recognition.

In order to learn spatio-temporal features, CNN-based approach [3] learns representations for RGB inputs using several temporal sampling. To better take the advantage of consecutive video frames, two-stream Network [5] proposed two separate CNNs for appearance and motion information respectively. Based on these works, trajectory-pooled deep convolutional descriptor [6], very deep two-stream Networks [7] and many other variations have been proposed to enhance the representation power of video for classification. Another attempt of using CNN-based approach for video classification is 3D convolutional neural networks [1] which extends 2D convolutions to 3D spatio-temporal convolutions over a very short video clips to learn motion features from raw frames implicitly and then aggregate predictions at the video level. More recently, X. Wang et al. [6] considered an action as a sequence of transformation across frames or key frames and employed two-stream siamese network to further improve classification performance.

RNNs have also shown its effectiveness for action recognition. N. Srivastava et al. [8] used LSTM (Long short term memory) in an unsupervised manner for better understanding of video representation. Ng et al. [9] extracted features with a two-stream networks and employed long short term memory fusion for action recognition. LRCN [10] also try to learn spatio-temporal features over long time periods aggregating powerful CNN features to discover long-range temporal relationships.

Despite its success in action recognition, many of these works largely depend on tremendous amount of training data and are still hard to train efficiently. In this work, we address this issue and propose to learn action representation in a non-parametric way.

2) One-shot learning: One-shot learning has been studied based on generative models. In early studies, bayesian algorithms were used to infer proper parameters with global prior distribution [11]. Hierarchical bayesian model [12] was also used with supercategory-level prior. A different approach is to learn an embedding space to incorporate metric learning such as nearest-neighbor for classification. Deep convolutional siamese [13] networks were proposed to classify pairs according to distance. Moreover, training is performed with triplet loss [14], [15] to enforce a distance ranking. Recently, Santoro et al. [4] proposed to adapt external memory for one-shot learning. Lastly, the most relevant work of this paper is a

matching network [16] which provides a novel framework amenable for one-shot learning using both external memory and metric learning. Our work is highly inspired by the matching network, then we adapt it to action recognition problems with image sequences.

III. Model

A. Overview

The overall structure of our proposed model is illustrated in Figure 2. First of all, we extract an image representation of each frame from video clips using pre-trained CNN algorithms to obtain input data X. We separate our input data X into two exclusive sets: support set S and input set X'. We sample a few clips from the input dataset $X = \{(x_i, y_i)\}_{i=1}^{n}$ as the support set which would act like matched examples. Then we store the support set S in an external long-term memory. Our model consists of two embedding networks for these two sets, support set S and input set X', respectively. Moreover, both embedding networks have their own external memory, the short-term memory for the input set X' and the long-term memory for the support set S.

The first network is used as an embedding of support set S. Each data in the support set has frame sequences from a clip so that we get embedding representations using GRU (Gated Recurrent Unit) [17] network. Moreover, we use those representations to control matching relations between the support set S and the input set X'. To enable this, we put those representations into the second embedding network as a context to be attended.

In the second embedding network, we build a memory based embedding structure to get representations from the input set X'. In the memory based embedding, representations are obtained from activated combinations of the current frame and the past frames. Then we sequentially embed those representations over a GRUs network and average over outputs of the GRUs network.

Different from the conventional classification model, the last part of our model is not a fully connected neural network. We combine a metric calculation module based on cosine similarity between the input set X' and the support set S. After summing up similarities, we can get the probabilities of target classes.

B. Memory based Embedding

In our model, we use memory network [18] to get representations from input video frames X'. Memory network is a kind of memory augmented neural network [18] which recently showed promising performances at various tasks. Memory network can reinforce an input representation from matched items in an external memory. We use a main part of memory network to obtain enhanced representations of input frames sequences. However, a main difference between memory network and our model is that we sequentially add previous scenes in a clip into a short-term memory.

Figure 3 shows the memory-based embedding network (MEN). In MEN, as sequences are fed into the network, we

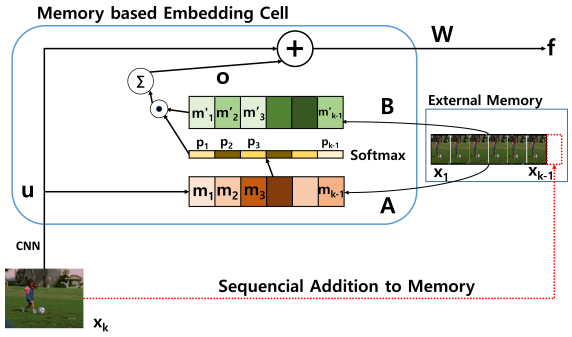

Fig. 3. Illustration of a memory-based embedding unit. Previous scenes which were sequentially added into external memory are embedded by the matrix A and B. The first embedded matrix M and the input embedding u generate probability vector p followed by the softmax function. Then the vector o is a sum over the second embedded matrix M' weighted by the vector p. Finally, the vector o and u are added, then we obtain the final output f passed through W.

stack previous sequences $x_1, x_2, ..., x_{k-1}$ in a short-term memory. Then we convert those frames in a short-term memory into a transformed memory $M = [m_1, m_2, ..., m_{k-1}]$. More specifically, each $x_i (i = 1...k-1)$ is converted into $m_i \in R^d$ computed by embedding matrix A $(i.e, m_i = Ax_i)$. Input x_k is also embedded to obtain and vector $u = Cx_k$. In the embedding space, we compute the matches between the input frame u and the memories $m_1, m_2, ..., m_{k-1}$ by taking the inner product followed by the softmax:

$$p_i = \text{softmax}(u^T m_i) \tag{1}$$

where $\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$. Here, j is the index of items in the short-term memory M. Therefore $p = [p_1, ..., p_{k-1}]$ can be considered as a matched probability vector over the past frame sequences.

We use the distribution p as an attention coefficient for each memory. In order to use those probabilities, we also compute another transformed memories $M' = [m'_i, ..., m'_{k-1}]$ using an embedding matrix B $(i.e, m'_i = Bx_i)$. The output vector o is then a sum over the M', weighted by distribution from p:

$$o = \sum_i p_i m'_i \tag{2}$$

Then we add the output vector o and the input embedding u to pass through a final weight matrix W:

$$f = W(o + u) \tag{3}$$

At last, combined representation f is fed into the GRU layer to obtain the final input representation g as shown in Figure 2.

C. Gated Recurrent Unit with Content-based attention

For both of the input set embedding and the support set embedding, we use the GRUs to get representations from the sequences. While vanilla RNNs suffer from the well-known vanishing and exploding gradient problems [19], the

LSTM [20] was proposed to overcome those issues. The GRU which is a variant of the LSTM, was found to achieve better performance than the LSTM on some tasks [21]. The vanilla GRU is defined by the following equations:

$$r_t = \text{sigm}(W_{xr}x_t + W_{hr}h_{t-1} + b_r) \tag{4}$$

$$z_t = \text{sigm}(W_{xz}x_t + W_{hz}h_{t-1} + b_z) \tag{5}$$

$$\bar{h}_t = \tanh(W_{xh}x_t + W_{hh}(r_t \odot h_{t-1}) + b_h) \tag{6}$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \bar{h}_t \tag{7}$$

In these formulation, the x_t is input at time t and the h_{t-1} is a hidden activation at time $t-1$ which is iteratively fed into the GRU cell. The W variables are the weight matrices and the b variables are the biases. The operation $\odot$ denotes the element-wise vector product. The update gate z_t balances between previous activation h_{t-1} and the candidate activation $\bar{h}_t$. The reset gate r_t allows it to forget the previous state.

While vanilla GRUs work for the support set embedding, we adapt a content-based attention guided by the support set to GRUs in the input set embedding. Since the classification strategy depends on the support set, it can be beneficial to make the input embedding conditioned by the support set. We can obtain a content-based attention c_t from dot-product activation between a previous hidden state h_{t-1} at time t and the support set S:

$$c_t = \sum_{i=1}^{|S|} a(h_{t-1}, s_i)s_i \tag{8}$$

$$a(h_{t-1}, s_i) = \text{softmax}(h_{t-1}^T s_i) \tag{9}$$

Then we get a reinforced hidden state $h_{t-1} = [h_{t-1}, c_t]$ which gets concatenated with a content-based attention c_t.

D. Metric based classification

In general, the last layer of a deep learning model is a fully-connected layer designed to classify target labels. The last layer produces probabilities of all target labels through the softmax function but it needs a lot of parameters and easily gets overfitted. In order to optimize parameters in a fully-connected layer, we should prepare a huge amount of data, especially for complex problems such as human action recognition. Therefore in our work, we eliminate the fully-connected layer just before the softmax function.

Instead, we use a differentiable nearest neighbor layer which is based on a cosine similarity metric. We first remind support set S from the external long-term memory. A similarity $\hat{a}_i$ of the input representation vector g against each support set s_i is calculated as:

$$\hat{a_i} = \frac{g \cdot s_i}{\|g\|\|s_i\|} \tag{10}$$

The similarity $\hat{a}_i$ also means a inverse distance between input representation and each support sample in the support set embedding space. We can use $\hat{a}_i$ as a measure of the similarity in a nearest neighbor classifier. However, in our model, we can generalize our method with classes in the support set as below.

When we stack the support set $S = \{(s_i, l_i)\}_{i=1}^m$ in the memory, we also store one-hot vectors of support set's labels l_i. Then we can easily obtain a sum of similarities by multiplying the similarity vector $\hat{a}_i$ and the one-hot vector l_i:

$$\bar{y} = \sum_i \hat{a} l_i \tag{11}$$

The vector $\bar{y}$ is the sum of cosine similarities which are already normalized between -1 and 1, which allows to get distribution of closeness in the vector space by taking the softmax function:

$$\hat{y} = \text{softmax}(\bar{y}) \tag{12}$$

Despite using metric calculation, we can train our model in an end-to-end fashion. Moreover, since we use not only the non-parametric method but also parametric algorithms such as GRUs, we should optimize an objective function to train parameters in our model. More precisely, our training objective is as follows:

$$\theta = \arg\max_\theta \sum_{X'} P_\theta(y_i'|x_i', S) \tag{13}$$

where $X' = \{(x_i', y_i')\}_{i=1}^k$.

IV. Experiments

A. Datasets

We perform experiments on three different datasets: UCF-11 [22], UCF sport [23] and HMDB 51 [24]. Compared to recently released datasets such as UCF-101, those datasets have fewer classes and smaller amount of videos. However, since we intended to show that our model could be trained with small-sized data and to deal with extending target classes, those datasets are considered suitable for such purposes.

1) UCF-11 Human Action Dataset: [22] is a challenging dataset which has following properties: 1) a mix of steady cameras and shaky cameras, 2) cluttered background, 3) variation in object scale, 4) varied viewpoint, 5) varied illumination, and 6) low resolution. Dataset contains 11 categories and 1,600 sequences in total. Action categories include: basketball shooting, biking/cycling, diving, golf swinging, horse back riding, soccer juggling, swinging, tennis swinging, trampoline jumping, volleyball spiking, and walking with a dog. Similar to the original setup, we used cross-validation for a pre-defined set of 25 folds. Performance is calculated by the average accuracy over all classes.

2) UCF Sport Dataset: [23] includes a total of 150 sequences with the resolution of 720 x 480. The dataset contains 10 sport action categories: swinging, diving, kicking ball, weight lifting, horse riding, running, skateboarding, swinging (on the bench), golf swinging and walking. It is also a challenging dataset which includes various actions featured in a wide range of scenes and view points. The training setup was recommended in Rodriguez et al. [23], using a Leave one out cross validation (LOOCV) scheme. However, we randomly separate 10 train/test splits and report the average accuracy over best performances.

TABLE I
COMPARISON OF PERFORMANCE ON UCF-11, UCF SPORT AND HMDB-51 WITH THE STATE-OF-THE-ART MODELS. NOTE THAT BASELINE METHODS OF HMDB 51 ONLY USE RGB DATA AS OURS.

UCF-11 Human Action		**UCF Sport**		**HMDB 51 (RGB Only)**	
Method	Video Acc.	Method	Video Acc.	Method	Video Acc.
Ikizler-Cinbis et al. [25]	75.2%	Kovashka et al. [26]	87.2%	Simonyan et al. [5]	40.5%
Wang et al. [27]	84.2%	Wang et al. [27]	88.2%	Sharma et al. [28]	41.3%
Jungchan Cho et al [29]	86.1%	Jungchan Cho et al [29]	**89.7**%	Srivastava et al. [8]	**44.0**%
Our Model(1-shot)	90.5%	Our Model(1-shot)	87.0%	Our Model(1-shot)	31.6%
Our Model(5-shot)	**91.7%**	Our Model(5-shot)	89.0%	Our Model(5-shot)	38.8%

3) HMDB51 Dataset: [24] contains 51 distinct action categories and 6,766 video clips extracted from a wide range of sources such as movies and YouTube. Each category includes at least 101 samples. We follow the original evaluation setup which provided three train/test split. For every class and split, there are 70 videos for training and 30 videos for testing. We compute the average accuracy over the three splits as a performance measure.

B. Experimental Setting

We perform two kinds of experiments with the proposed model: supervised learning and one-shot learning. The first experiment is to compare the performances of the proposed model with state-of-the-art methods. In the second experiment, we set up an experimental environment for the one-shot learning framework to predict unseen classes.

1) Supervised Learning: The first experiment is designed on a supervised learning framework. On the three datasets, we evaluate the classification accuracy of action classes. Then we compare our models with a set of baselines proposed recently on those datasets.

For image representations, we use the VGGNet [30] to extract features from a final fully-connect network. Before extracting features which have 4096 length of vectors, we split a video to several clips to fix the length of sequences for the sake of implementation. In this experiment, each clip consists of 16 frames and each video has several clips depending on the length of the video. However, basically variable length of clips can be used in our framework.

The support set S is sampled from the input set X and we set the number of samples per class as one or five. For example, if we set the per-class sample size as five on UCF-11 dataset, we could collect 55 samples from the input set. Then we store the support set into a long-term memory and also use the support set from the long-term memory in the test phase.

For evaluation, we calculated the accuracy on a video from the accuracy on clips. At test phase, we computed the class prediction for each clip. Since a video consists of several clips, we averaged predictions of clips to obtain the accuracy of the entire video.

In this experiment, a model is trained using a stochastic gradient decent optimizer. The initial learning rate is set to be 0.001 and the decay ratio is adapted as 0.95 per 500 iterations. We use dropout when we stack past frames into external memory slots.

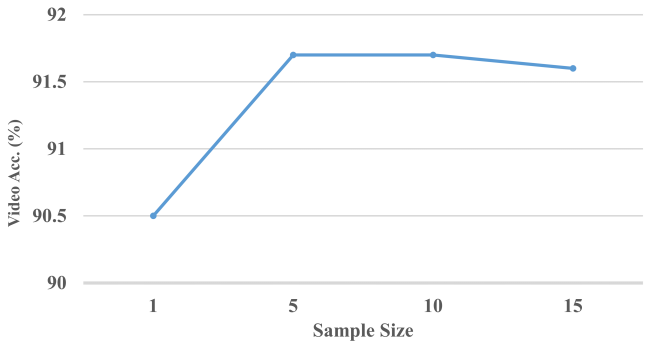

Fig. 4. Experimental results of the supervised learning on UCF-11 action dataset; video accuracy scores under changes in sample size per class of support set

2) One-shot learning: For one-shot learning, to predict unseen classes, we set up environment as followings. First, we split target classes Y into two disjoint groups: training classes L, test classes L'. While training classes L are used for training our model, test classes L' are not observed. Then we can set this problem P as N-way classification which means the number of target classes is N. For training, we first sample N classes which denote $\hat{L}$ from training classes L per a mini-batch and also sample support set S and input set X in $\hat{L}$. For instance, if the number of support set for each class is s, we sample $N \times s$ examples for N chosen classes in a mini-batch. After training is completed, we can test our model only within test classes L' for predicting unseen classes. While the test strategy is the same as the training strategy, we sample N classes which denote $\hat{L}'$ from test classes L'. Then we sample support set S' and input set X' in $\hat{L}'$. For both processes, note that we create a different support set respectively.

Labels of sampled classes are set from 0 to $N-1$ for N-way classification so that each class number has no meaning. This strategy prevents the model to memorize the order of classes. Therefore, this can endow the embedding mechanism with the power of generalization to convert any classes into an appropriate vector in the embedding space. We used UCF-11 dataset and HMDB 51 dataset for the one-shot learning experiment. In this experiment, we did not conduct LOOCV

TABLE II
RESULTS OF ONE-SHOT LEARNING ON UCF-11 ACTION DATASET WITH BASELINES

Method	1-shot Clip Acc.	5-shot Clip Acc.
Baseline(Random)	20.0%	20.0%
Baseline(FC)	20.8%	20.1%
Baseline(Pixel)	23.9%	25.7%
Our Model	**42.1%**	**50.8%**

TABLE III
RESULTS OF ONE-SHOT LEARNING ON HMDB 51 ACTION DATASET WITH BASELINES

Method	1-shot Clip Acc.	5-shot Clip Acc.
Baseline(Random)	20.0%	20.0%
Baseline(FC)	20.7%	20.3%
Baseline(Pixel)	24.4%	27.4%
Our Model	**42.9%**	**58.0%**

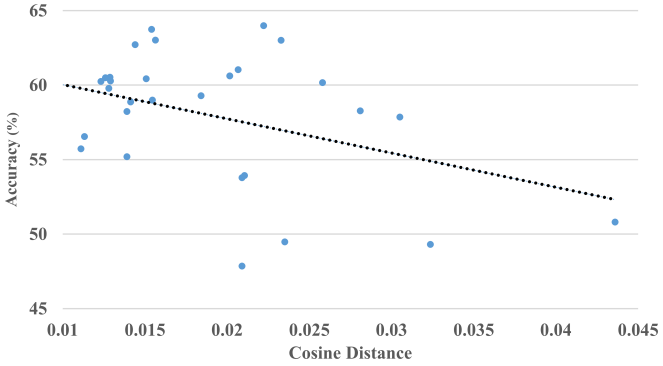

Fig. 5. Experimental results of 5-shot learning on HMDB 51 dataset; Video accuracy scores are inversely proportional to distances between train classes and test classes

or cross-validation method. Only 5-way classification was performed and both 1 shot and 5 shot of support set were used for experiments.

For UCF-11 dataset, we randomly split 11 classes into 6 training classes and 5 test classes. On HMDB 51 dataset, we randomly split 41 training classes and 10 test classes. In this experimental setting, the correlation between train classes L, test classes L' can be crucial. More correlated splits can show better performances due to the similar topology in a vector space. Therefore we made 15 class-splits randomly for each experiment and averaged best accuracies of all trials. Other conditions for experiments are almost same as those for supervised learning.

C. Experimental Results

1) Supervised Learning: Table 1 shows accuracies on three datasets. Here, 'shot' (e.g., 1-shot, 5-shot) means the number of examples per class in the support set. On UCF-11, a relatively smaller set, our model outperforms a set of baselines proposed recently. These results reveals our metric based classification strategy has big advantages on datasets including a small amount of data. Obviously, 5-shot MVN works better than 1-shot because using more examples helps the model.

On HMDB 51 action dataset, our model shows comparable results to recent works which only use RGB data as our method did. In metric classification, classifying many classes needs more comparison with examples. Hence, we obtained lower performances than small datasets. And 5-shot model also shows better performance than the 1-shot model.

Furthermore, we changed the size of sample per class to understand how it effects performances. We performed those experiments only on UCF-11 action dataset. Figure 4 shows that increasing the number of samples per class can improve recognition accuracy. However, the ratio of performance growth drops from the sample size of 5. From this, we can conclude that too many samples do not give sufficient advantages. Since the support set shares model capacities with parameters in the model, too many samples in the support set cause overfitting due to the high complexity.

2) One-shot learning: For comparison, we considered three kinds of baselines. First, we performed 5-way classification experiments so that the random baseline can be 1/5 (20%). Second baseline adapted a fully connected layer instead of our nearest neighbor layer. The number of hidden nodes in this layer is the same as the number of GRUs. Then with this baseline, we can compare metric based classification with conventional method. Lastly, we proposed third baseline which only used raw pixel inputs to eliminate the representation power of VGGNet. Each raw frame was resized to a 64x64 image and flattened to a 4096 length vector to have the same size as the inputs from VGGNet. Note that the same structure as our model was used for the third baseline.

Table 2 compares performances of our model with baselines. In this experiment, clip accuracy is used as performance measure instead of video accuracy. Compared with baseline with fully-connected layer (FC), our model shows better performance. In this experimental setting to predict unseen classes, the fully connected layer has a big disadvantage because of memorizing class information with parameters. In contrast, nearest neighbor method only consider similarities between vectors so that metric based classification can be more suitable to this experiment. Third baseline shows better performance than random baseline. However, compared with our method, the result reveals that representation power of VGGNet sufficiently influence performance.

Table 3 demonstrates that our model outperformed over baselines. Although HMDB 51 action dataset has more classes than UCF-11 action dataset, results shows similar scores as previous experiment. Since we fixed 5 way classification, the size of dataset would not crucially influence performances. However, in both train and test phase, classes are sampled from more classes so that model can see more various combination of classes. Therefore, this could have led to better performances compared to the previous experiment in Table 2.

We conducted 15 trials per each experiment due to the correlation issue between train classes and test classes. Figure 5. shows how distances between train classes and test classes

in the HMDB 51 dataset affect performances in 30 trials. We used a cosine distance metric between centroid vectors of two splits. As distances between train classes and test classes are larger, performances seem to be worse. We can conclude that the one-shot learning can show good results for unseen classes which are highly related to train classes as human can easily generalize new concepts which are more familiar with.

V. Conclusion

In this paper, we proposed a novel model which improves the performance on action recognition dataset with a small amount of data. With the metric based classifier and the external memory, we can train our model in a framework that combines parametric networks in a non-parametric manner. Therefore our model achieved state-of-the-art results on small datasets and a comparable result on a larger dataset.

Furthermore, we tested our model on one-shot learning problem. We can predict unseen classes from few examples in the external memory based on a novel training strategy. Compared to baselines, experimental results show improved performances on various datasets.

Acknowledgment

This work was supported by Ministry of Culture, Sports and Tourism (MCST) and Korea Creative Content Agency (KOCCA) in the Culture Technology (CT) Research and Development Program 2016.

References

[1] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," *arXiv preprint arXiv:1412.0767*, 2014.

[2] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.

[3] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *CVPR*, 2014.

[4] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "One-shot learning with memory-augmented neural networks," *arXiv preprint arXiv:1605.06065*, 2016.

[5] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems*, 2014, pp. 568–576.

[6] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4305–4314.

[7] L. Wang, Y. Xiong, Z. Wang, and Y. Qiao, "Towards good practices for very deep two-stream convnets," *arXiv preprint arXiv:1507.02159*, 2015.

[8] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015, pp. 843–852.

[9] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4694–4702.

[10] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2625–2634.

[11] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 4, pp. 594–611, 2006.

[12] R. Salakhutdinov, J. Tenenbaum, and A. Torralba, "One-shot learning with a hierarchical nonparametric bayesian model," *Unsupervised and Transfer Learning Challenges in Machine Learning, Volume 7*, p. 225, 2012.

[13] G. Koch, "Siamese neural networks for one-shot image recognition," Ph.D. dissertation, University of Toronto, 2015.

[14] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 815–823.

[15] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Web-scale training for face identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2746–2754.

[16] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," *arXiv preprint arXiv:1606.04080*, 2016.

[17] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[18] S. Sukhbaatar, J. Weston, R. Fergus *et al.*, "End-to-end memory networks," in *Advances in neural information processing systems*, 2015, pp. 2440–2448.

[19] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[22] J. Liu, J. Luo, and M. Shah, "Recognizing realistic actions from videos in the wild," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 1996–2003.

[23] M. D. Rodriguez, J. Ahmed, and M. Shah, "Action mach a spatio-temporal maximum average correlation height filter for action recognition," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[24] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "Hmdb: a large video database for human motion recognition," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 2556–2563.

[25] N. Ikizler-Cinbis and S. Sclaroff, "Object, scene and actions: Combining multiple features for human action recognition," in *European conference on computer vision*. Springer, 2010, pp. 494–507.

[26] A. Kovashka and K. Grauman, "Learning a hierarchy of discriminative space-time neighborhood features for human action recognition," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 2046–2053.

[27] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Action recognition by dense trajectories," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 3169–3176.

[28] S. Sharma, R. Kiros, and R. Salakhutdinov, "Action recognition using visual attention," *arXiv preprint arXiv:1511.04119*, 2015.

[29] J. Cho, M. Lee, H. J. Chang, and S. Oh, "Robust action recognition using local motion and group sparsity," *Pattern Recognition*, vol. 47, no. 5, pp. 1813–1825, 2014.

[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.