RNN Fisher Vectors for Action Recognition and Image Annotation

Guy Lev[1,2], Gil Sadeh[1], Benjamin Klein[1(✉)], and Lior Wolf[1]

[1] The Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv, Israel
beni.klein@gmail.com
[2] IBM Research, Haifa, Israel

Abstract. Recurrent Neural Networks (RNNs) have had considerable success in classifying and predicting sequences. We demonstrate that RNNs can be effectively used in order to encode sequences and provide effective representations. The methodology we use is based on Fisher Vectors, where the RNNs are the generative probabilistic models and the partial derivatives are computed using backpropagation. State of the art results are obtained in two central but distant tasks, which both rely on sequences: video action recognition and image annotation. We also show a surprising transfer learning result from the task of image annotation to the task of video action recognition.

Keywords: Action recognition · Image annotation · Fisher vectors · Recurrent Neural Networks

1 Introduction

Fisher Vectors have been shown to provide a significant performance gain on many different applications in the domain of computer vision [1–4]. In the domain of video action recognition, Fisher Vectors and Stacked Fisher Vectors [2] have recently outperformed state-of-the-art methods on multiple datasets [2,5]. Fisher Vectors (FV) have also recently been applied to word embedding (e.g. word2vec [6]) and have been shown to provide state of the art results on a variety of NLP tasks [7], as well as on image annotation and image search tasks [8].

In all of these contributions, the FV of a set of local descriptors is obtained as a sum of gradients of the log-likelihood of the descriptors in the set, with respect to the parameters of a probabilistic mixture model that was fitted on a training set in an unsupervised manner. Despite being richer than the mean vector pooling method, Fisher Vectors based on a probabilistic mixture model are invariant to order. This makes them less appealing for annotating, for example, video, in which the sequence of events determines much of the meaning.

This work presents a novel approach for FV representation of sequences using a Recurrent Neural Network (RNN). The RNN is trained to predict the next

Electronic supplementary material The online version of this chapter (doi:10.1007/978-3-319-46466-4_50) contains supplementary material, which is available to authorized users.

B. Leibe et al. (Eds.): ECCV 2016, Part VI, LNCS 9910, pp. 833–850, 2016.
DOI: 10.1007/978-3-319-46466-4_50

element of a sequence given the previous elements. Conveniently, the gradients needed for the computation of the FV are extracted using the available back-propagation infrastructure.

The new representation is sensitive to ordering and, therefore, mitigates the disadvantage of using the standard Fisher Vector representation. It is applied to two different and challenging tasks: video action recognition and image annotation by sentences.

Several recent works have proposed to use an RNN for sentence representation [9–12]. The Recurrent Neural Network Fisher Vector (RNN-FV) method differs from these works in that a sequence is represented by using derived gradient from the RNN as a vector representation, instead of using a hidden or an output layer of the RNN.

The paper explores training an RNN regressor to predict the vector representation of the next element of a sequence given the previous ones (i.e. treating it as a regression task). In the image annotation and image search tasks, word embeddings are used for representing words. In the video action recognition task, the VGG [13] Convolutional Neural Network (CNN) is used to extract features from the frames of the video and the RNN is trained to predict the embedding of the next frame given the previous ones. Similarly, C3D [14] features of sequential video sub-volumes are used with the same training technique.

Although the image annotation and video action recognition tasks are quite different, a surprising boost in performance in the video action recognition task was achieved by using a transfer learning approach from the image annotation task. Specifically, the VGG image embedding of a frame is projected using a linear transformation which was learned on matching images and sentences by the Canonical Correlation Analysis (CCA) algorithm [15].

The proposed RNN-FV method achieves state-of-the-art results in action recognition on the HMDB51 [16] and UCF101 [17] datasets. In the image annotation and image search tasks, the RNN-FV method is used for the representation of sentences and achieves state-of-the-art results on the Flickr8K dataset [18] and competitive results on other benchmarks.

2 Previous Work

Action Recognition. As in other object recognition problems, the standard pipeline in action recognition is comprised of three main steps: feature extraction, pooling and classification. Many works [19–21] have focused on the first step of extracting local descriptors. Laptev et al. [22] extend the notion of spatial interest points into the spatio-temporal domain and show how the resulting features can be used for a compact representation of video data. Wang et al. [23,24] used low-level hand-crafted features such as histogram of oriented gradients (HOG), histogram of optical flow (HOF) and motion boundary histogram (MBH).

Recent works have attempted to replace these hand-crafted features by deep-learned features for video action recognition due to its wide success in the image domain. Early attempts [25–27] achieved lower results in comparison to hand-crafted features, proving that it is challenging to apply deep-learning techniques on

videos due to the relatively small number of available datasets and complex motion patterns. More recent attempts managed to overcome these challenges and achieve state of the art results with deep-learned features. Simonyan et al. [28] designed two-stream ConvNets for learning both the appearance of the video frame and the motion as reflected by the estimated optical flow. Du Tran et al. [14] designed an effective approach for spatiotemporal feature learning using 3-dimensional ConvNets.

In the second step of the pipeline, the pooling, Wang et al. [29] compared different pooling techniques for the application of action recognition and showed empirically that the Fisher Vector encoding has the best performance. Recently, more complex pooling methods were demonstrated by Peng et al. [2] who proposed Stacked Fisher Vectors (SFV), a multi-layer nested Fisher Vector encoding and Wang et al. [5] who proposed a trajectory-pooled deep-convolutional descriptor (TDD). TDD uses both a motion CNN, trained on UCF101, and an appearance CNN, originally trained on ImageNet [30], and fine-tuned on UCF101. Fernando et al. [31] suggested to capture the temporal ordering of a particular video by training a linear ranking machine on the frames of that video. The parameters of the ranking machine are used as the video representation for action recognition. In parallel to our work, Nagel et al. [32] proposed using event Fisher Vectors for encoding a visual stream. They considered two different generative models beyond the Gaussian Mixture Model. The first is the Student's-t mixture model which has heavy tails but is not sensitive to the order of the elements in the sequence. The second is the Hidden Markov Model which can capture the temporal ordering of the elements in the sequence. Our work is using a Fisher Vector which is defined on a Recurrent Neural Network model.

Image Annotation and Image Search. In the past few years, the state-of-the-art results in image annotation and image search have been provided by deep learning approaches [8,33–41]. A typical system is composed of three important components: (i) Image Representation, (ii) Sentence Representation, and (iii) Matching Images and Sentences. The image is usually represented by applying a pre-trained CNN on the image and taking the activations from the last hidden layer.

There are several different approaches for the sentence representation; Socher et al. [33] used a dependency tree Recursive Neural Network. Yan et al. [34] used a TF-IDF histogram over the vocabulary. Klein et al. [8] used word2vec [6] as the word embedding and then applied Fisher Vector based on a Hybrid Gaussian-Laplacian Mixture Model (HGLMM) in order to pool the word2vec embeddings of the words in a given sentence into a single representation. Ma et al. [41] proposed a matching CNN (m-CNN) that composes words to different semantic fragments and learns the inter-modal relations between the image and the composed fragments at different levels.

Since a sentence can be seen as a sequence of words, many works have used a Recurrent Neural Network (RNN) in order to represent sentences [12,35–37,40]. To address the need for capturing long term semantics in the sentence, these works mainly use Long Short-Term Memory (LSTM) [42] or Gated Recurrent Unit (GRU) [43] cells. Generally, the RNN treats a sentence as an ordered sequence of

words, and incrementally encodes a semantic vector of the sentence, word-by-word. At each time step, a new word is encoded into the semantic vector, until the end of the sentence is reached. All of the words and their dependencies will then have been embedded into the semantic vector, which can be used as a feature vector representation of the entire sentence. Our work also uses an RNN in order to represent sentences, but takes the derived gradient from the RNN as features, instead of using a hidden or an output layer of the RNN. In parallel to our work, Gordo et al. [44] proposed using the gradient representation of CNNs for images.

A number of techniques have been proposed for the task of matching images and sentences. Klein et al. [8] used CCA [15] and Yan et al. [34] introduced a Deep CCA in order to project the images and sentences into a common space and then performed a nearest neighbor search between the images and the sentences in the common space. Kiros et al. [37], Karpathy et al. [35], Socher et al. [33] and Ma et al. [41] used a contrastive loss function trained on matching and unmatching pairs of (image, sentence) in order to learn a score function for a given pair. Mao et al. [36] and Vinyals et al. [40] learned a probabilistic model for inferring a sentence given an image and, therefore, are able to compute the probability that a given sentence will be created by a given image and used it as the score.

Related Work. [45,46] have also proposed methods incorporating advanced pooling techniques within the CNN and backpropagation infrastructure.

2.1 Baseline Pooling Methods

In this section, we describe two baseline pooling methods that can represent a multiset of vectors as a single vector. The notation of a multiset is used to clarify that the order of the vectors does not affect the representation, and that a vector can appear more than once. Both methods can be applied to sequences. However, the resulting representation will be insensitive to ordering.

Mean Vector. This pooling method takes a multiset of vectors, $X = \{x_1..x_N\} \in \mathbb{R}^D$, and computes its mean: $v = \frac{1}{N}\sum_{i=1}^{N} x_i$. Clearly, the vector v that results from the pooling is in $\mathbb{R}^D$.

Fisher Vector of a GMM. Given a multiset of vectors, $X = \{x_1..x_N\} \in \mathbb{R}^D$, the standard FV [47] is defined as the gradient of the log-likelihood of X with respect to the parameters of a pre-trained Diagonal-Covariance Gaussian Mixture Model (GMM). In [4], Perronnin et al. introduced two normalizations of the FV which improved its performance. It is worth noting that the linear structure of the GMM FV pooling would not be preserved in the RNN model, where the probability of an element in the sequence depends on all the previous elements.

3 RNN-Based Fisher Vector

The pooling methods described above share a common disadvantage: insensitivity to the order of the elements in the sequence. A way to tackle this, while keeping the

power of gradient-based representation, would be to replace the Gaussian model by a generative sequence model that takes into account the order of elements in the sequence. A desirable property of the sequence model would be the ability to calculate the gradient (with respect to the model's parameters) of the likelihood estimate by this model to an input sequence.

In this section, we show that such a model can be obtained by training an RNN regressor to predict the embedding of the next element in a sequence, given the previous elements. Having this, we propose, for the first time, the RNN-FV: A Fisher Vector that is based on such an RNN sequence model.

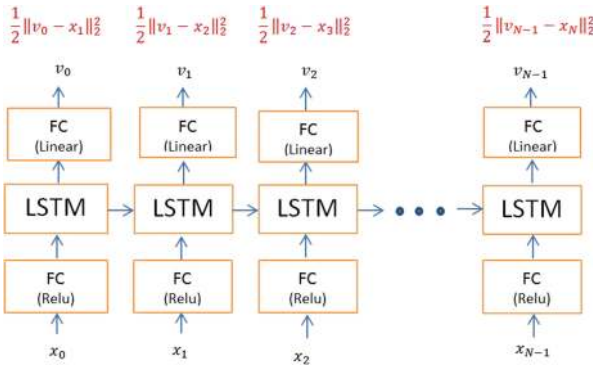

Fig. 1. RNN structure and loss function (in red), as was trained for the action recognition task. The RNN is trained to predict the next element of the sequence, given the previous ones. The gradient of the loss function (which can be seen as likelihood), with respect to the RNN's weights, constitutes the unnormalized RNN-FV. (Color figure online)

Given a sequence of vectors S with N vector elements $x_1, ..., x_N \in \mathbb{R}^D$, we convert it to the input sequence $X = (x_0, x_1, ..., x_{N-1})$, where $x_0 = x_{start}$. This special element is used to denote the beginning of the input sequence, and we use $x_{start} = 0$ throughout this paper. The RNN is trained to predict, at each time step i, the next element x_{i+1} of the sequence, given the previous elements $x_0, ..., x_i$. Therefore, given the input sequence, the target sequence would be: $Y = (x_1, x_2, ...x_N)$. The training data and the training process are application dependent, as described in Sect. 4 for action recognition and in Sect. 5 for image annotation. There are several regression loss functions that can be used. Here, we consider the following loss function:

$$Loss(y, v) = \frac{1}{2}\|y - v\|^2 \tag{1}$$

where y is the target vector and v is the predicted vector.

After the RNN training is done, and given a new sequence S, the derived sequence X is fed to the RNN. Denote the output of the RNN at time step i

$(i = 0, ..., N-1)$ by $RNN(x_0, ..., x_i) = v_i \in \mathbb{R}^D$. The target at time step i is x_{i+1} (the next element in the sequence), and the loss is:

$$Loss(x_{i+1}, v_i) = \frac{1}{2}\|x_{i+1} - v_i\|^2 \tag{2}$$

The RNN can be seen as a generative model, and the likelihood of any vector x being the next element of the sequence, given $x_0, ..., x_i$, can be defined as:

$$p\left(x|x_0, ..., x_i\right) = (2\pi)^{-D/2} \exp\left(-\frac{1}{2}\|x - v_i\|^2\right) \tag{3}$$

Here, we are interested in the likelihood of the correct prediction, i.e., in the likelihood of the vector x_{i+1} given $x_0, ..., x_i$: $p\left(x_{i+1}|x_0, ..., x_i\right)$.

The RNN-based likelihood of the entire sequence X is:

$$p(X) = \prod_{i=0}^{N-1} p\left(x_{i+1}|x_0, ..., x_i\right) \tag{4}$$

The negative log likelihood of X is:

$$\begin{aligned}\mathcal{L}(X) &= -\log\left(p(X)\right) = -\sum_{i=0}^{N-1} \log\left(p\left(x_{i+1}|x_0, ..., x_i\right)\right) \\ &= \frac{ND}{2}\log(2\pi) + \frac{1}{2}\sum_{i=0}^{N-1}\|x_{i+1} - v_i\|^2\end{aligned} \tag{5}$$

In order to represent X using the Fisher Vector scheme, we have to compute the gradient of $\mathcal{L}(X)$ with respect to our model's parameters. With RNN being our model, the parameters are the weights W of the network. By (2) and (5), we get that $\mathcal{L}(X)$ equals the loss that would be obtained when X is fed as input to the RNN, up to an additive constant. Therefore, the desired gradient can be computed by backpropagation: we feed X to the network and perform forward and backward passes. The obtained gradient $\nabla_W \mathcal{L}(X)$ would be the (unnormalized) RNN-FV representation of X. Notice that this gradient is *not* used to update the network's weights as done in training - here we perform backpropagation *at inference time*. Other loss functions may be used instead of the one presented in this analysis. Given a sequence, the gradient of the RNN loss may serve as the sequence representation, even if the loss is not interpretable as a likelihood. Figure 1 illustrates the RNN structure and the loss function that we used for the action recognition task.

3.1 Normalization of the RNN-FV

It was suggested by [47] that normalizing the FVs by the Fisher Information Matrix is beneficial. We approximated the diagonal of the Fisher Information Matrix (FIM), which is usually used for FV normalization. Note, however, that we did not observe any empirical improvement due to this normalization, and our experiments are reported without it.

4 Action Recognition Pipeline

The action recognition pipeline contains the underlying appearance features used to encode the video, the sequence encoding using the RNN-FV, and an SVM classifier on top. The entire pipeline is illustrated in Fig. 2. In this section, we discuss each step of the pipeline.

4.1 Visual Features

The RNN-FV is capable of encoding the sequence properties, and as underlying features, we rely on video encodings that are based on single frames or on fixed length blocks of frames.

VGG. Using the pre-trained 19-layer VGG convolutional network [13], we extract a 4096-dimensional representation of each video frame. The VGG pipeline is used, namely, the original image is cropped in ten different ways into 224 by 224 pixel images: the four corners, the center, and their x-axis mirror image. The mean intensity is then subtracted in each color channel and the resulting images are encoded by the network. The average of the 10 feature vectors obtained is then used as the single image representation. In order to speed up the method, the input video was sub-sampled, and one in every 10 frames was encoded. Empirically, we noticed that recognition performance was not harmed by this sub-sampling. To further reduce run-time, the data dimensionality was reduced via PCA to 500D. In addition, L2 normalization was applied to each vector. All PCAs in this work were trained for each dataset and each training/test split separately, using only the training data.

CCA. Using the same VGG representation of video frames as mentioned above and the code of [8][1], we represented each frame by a vector as follows: we considered the common image-sentence vector space obtained by the CCA algorithm, using the best model (GMM+HGLMM) of [8] trained on the COCO dataset [48]. We mapped each frame to that vector space, getting a 4096-dimensional image representation. As the final frame representation, we used the first (i.e. the principal) 500 dimensions. For our application, the projected VGG representations were L2 normalized. The CCA was trained for an unrelated task of image to sentence matching, and its success, therefore, suggests a new application of transfer learning: from image annotation to action recognition.

C3D. While the representations above encode single frames, the C3D method [14] splits the video into sub-volumes that are encoded one by one. Following the recommended settings, we applied the C3D pre-trained 3D convolutional neural network in order to extract a 4096D representation of each 16-frame blocks. The blocks are sampled with an 8 frame stride. Following feature extraction, PCA dimensionality reduction (500D) and L2 normalization were applied. Notice that while we used the available pretrained C3D network, our results are not comparable to [14]'s highest reported performance which was

[1] Available at www.cs.tau.ac.il/~wolf/code/hglmm.

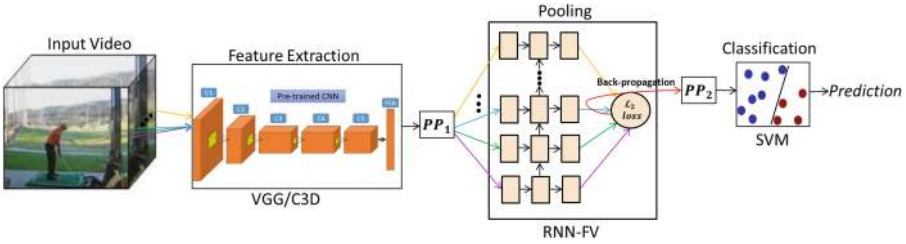

Fig. 2. Our general action recognition pipeline is composed of 6 steps: (a) Input Data - we use subsampled video frames or frame blocks as input to our system. (b) Feature Extraction - we extract features from the frames/frame-blocks using VGG/C3D pretrained CNN. (c) Post-Processing (PP_1) - PCA/CCA dimension reduction and L_2 normalization are performed. (d) Pooling - the extracted sequential features are fed into the RNN, then backpropagation is performed to obtain the partial derivatives with respect to the weights of the last fully-connected layer. (e) Post-Processing (PP_2) - PCA dimension reduction is performed, followed by power normalization and L_2 normalization. (f) Classification - the final representation is fed into a linear multi-class SVM classifier which predicts the estimated action label.

reached using an ensemble of 3 C3D networks (to our knowledge, the other two networks were not released) combined with idt [49].

4.2 Network Structure

Our RNN model (illustrated in Fig. 1) consists of three layers: a 200D fully-connected layer with Leaky-Relu activation ($\alpha = 0.1$), a 200D Long Short-Term Memory (LSTM) [42] layer, and a 500D linear fully-connected layer. Our network is trained as a regressor with the mean square error (MSE) loss function. Weight decay and dropouts were also applied. An improvement in recognition performance was noticed when the dropout rate was enlarged, up to a rate of 0.95, due to its ability to ensure the discriminative characteristics of each weight and hence also of each partial derivative in the gradient.

4.3 Training and Classification

We train the RNN to predict the next element in our video representation sequence, given the previous elements, as described in Sect. 3. In our experiments, we use only the part of gradient corresponding to the weights of the last fully-connected layer. Empirically, we saw no improvement when using the partial derivatives with respect to the weights of other layers. In order to obtain a fixed size representation, we average the gradients over all time steps. The gradient representation dimension is $500 \times 201{=}100500$, which is the number of weights in the last fully-connected layer. We then apply PCA to reduce the representation size to 1000D, followed by power and L2 normalization.

Video classification is performed using a linear SVM with a parameter $C = 1$. Empirically, we noticed that the best recognition performance is obtained

very quickly and hence early stopping is necessary. In order to choose an early stopping point, we use a validation set. Some of the videos in the dataset are actually segments of the same original video, and are included in the dataset as different samples. Care was taken to ensure that no such similar videos are in both the training and validation sets, in order to guarantee that high validation accuracy will ensure good generalization and not merely over-fitting.

After each RNN epoch, we extract the RNN-FV representation as described above, train a linear SVM classifier on the training set and evaluate the performance on the validation set. The early stopping point is chosen at the epoch with the highest recognition accuracy on the validation set. After choosing our model this way, we train an SVM classifier on all training samples (training + validation samples) and report our performance on the test set.

5 Image-Sentence Retrieval

In the image-sentence retrieval tasks (image annotation and image search), vector representations are extracted separately for the sentences and the images. These representations are then mapped into a common vector space, where the two are being matched. [8] have presented a similar pipeline for GMM-FV. We replace this representation with RNN-FV.

A sentence, being an ordered sequence of words, can be represented as a vector using the RNN-FV scheme. Given a sentence with N words $w_1, ..., w_N$, (where w_N is considered to be the period, namely a w_{end} special token), we treat the sentence as an ordered sequence $S = (w_0, w_1, ..., w_{N-1})$, where $w_0 = w_{start}$. An RNN is trained to predict, at each time step i, the next word w_{i+1} of the sentence, given the previous words $w_0, ..., w_i$. Therefore, given the input sequence S, the target sequence would be: $(w_1, w_2, ...w_N)$. The training data may be any large set of sentences. These sentences may be extracted from the dataset of a specific benchmark, or, in order to obtain a generic representation, any external corpus, e.g., Wikipedia, may be used.

As observed in the action recognition case, we did not benefit from extracting partial derivatives with respect to the weights of the hidden layers, and hence we only use those of the output layer as our representation.

The input to the network is the word's embedding, a 300D vector in our case, followed by an LSTM layer of size 100. The output layer is a fully-connected one, where the (300 dimensional) word embedding of the next word is predicted. We use no activation function at the output layer.

For matching images and text, each image is represented as a 4096-dimensional vector extracted using the 19-layer VGG, as described in Sect. 4.1. The regularized CCA algorithm [50], where the regularization parameter is selected based on the validation set, is used to match the the VGG representation with the sentence RNN-FV representation. In the shared CCA space, the cosine similarity is used in order to score (image, sentence) pairs.

We explored several configurations for training the RNN. **RNN training data** We employed either the training data of each split in the respective benchmark, or the 2010-English-Wikipedia-1M dataset made available by the Leipzig

Table 1. Pooling technique comparison: mean-pooling (MP), GMM-FV, RNN-FV, and their combinations with Temporal-Pyramid-Pooling (TPP), as evaluated on HMDB51 and UCF101 datasets. Three types of sequential features are used: VGG-PCA, VGG-CCA, and C3D. Additionally, a combination of descriptors (C3D + VGG) is evaluated, including a combination with idt GMM-FV [49]. All combinations are performed with early fusion. The table reports recognition average accuracy (higher is better).

HMDB51						
Method	MP	MP+TPP	GMM-FV	GMM-FV+TPP	RNN-FV	RNN-FV+TPP
VGG-PCA	42.16	46.14	36.8	38.54	45.62	47.38
VGG-CCA	43.05	47.19	39.61	41.5	46.14	46.01
C3D	51.2	54.01	45.82	48.54	52.88	53.51
C3D + VGG-CCA	37.1	56.23	50.19	52.16	54.33	55.77
C3D + VGG-CCA + idt	58.48	63.70	64.68	61.00	67.71	64.99
UCF101						
Method	MP	MP+TPP	GMM-FV	GMM-FV+TPP	RNN-FV	RNN-FV+TPP
VGG-PCA	75.51	77.34	76.53	77.12	79.29	81.56
VGG-CCA	77.49	78.68	76.84	77.95	79.49	80.83
C3D	81.05	81.72	80.04	80.10	82.33	82.81
C3D + VGG-CCA	65.55	87.85	86.73	87.11	88.01	88.09
C3D + VGG-CCA + idt	89.02	92.16	93.22	91.80	94.08	93.67

Corpora Collection [51]. This dataset contains 1 million sentences randomly sampled from English Wikipedia. **Word embedding** A word was represented either by word2vec, or by a "CCA word embedding" obtained as follows: (1) Each word was represented by the GMM+HGLMM FV representation of [8]. (2) These word representations were projected to the common image-sentence CCA space trained by [8] (on the respective dataset). (3) To reduce dimensionality, the first (i.e. the principal) 300 dimensions (out of 4096) of the mapped word representations were used. We made sure to match the training split according to the benchmark tested. **Sentence sequence direction** We explored both the conventional left-to-right sequence of words and the reverse direction.

We also explored using an RNN-FV which is based on a classifier RNN instead of a regressor. This design creates two challenges. The first is dimensionality: the size of the softmax layer equals the size of the dictionary, which is typically large. As a result, $\nabla_W \mathcal{L}(X)$ has a high dimensionality. The second issue is with generalization capability: since the softmax layer is fixed, a network cannot handle a sentence containing a word that does not appear in its training data. The RNN-FV regressor outperformed the RNN-FV classifier, and our experiments are reported without it.

6 Experiments

We evaluated the effectiveness of the various pooling methods on two important yet distinct application domains: action recognition and image textual annotation and search. As mentioned, applying the FIM normalization (Sect. 3.1) did

not seem to improve results. Another form of normalization we have tried, is to normalize each dimension of the gradient by subtracting its mean and dividing by its standard deviation. This also did not lead to an improved performance. Two normalizations that were found to be useful are the Power Normalization and the L2 Normalization, which were introduced in [52]. Both are employed, using a constant $\alpha = 1/2$. In addition to the experimental details provided in this section, further technical details and comparisons with baselines are given in the supplementary material.

6.1 Action Recognition

Our experiments were conducted on two large action recognition benchmarks. The UCF101 [17] dataset consists of 13,320 realistic action videos, collected from YouTube, and divided into 101 action categories. We use the three splits provided with this dataset in order to evaluate our results and report the average accuracy over these splits. The HMDB51 dataset [16] consists of 6766 action videos, collected from various sources, and divided into 51 action categories. Three splits are provided as an official benchmark and are used here. The average accuracy over these splits is reported.

We compare the performance of the RNN-FV to the baselines of mean-pooling and GMM-FV when combined with Temporal-Pyramid-Pooling (TPP) in order to validate that it is able to better capture temporal ordering information, as shown in Table 1. Three sets of features, as described in Sect. 4.1, are used: VGG coupled with PCA, VGG projected by the image to sentence matching CCA, and C3D.

As can be seen in Table 1, the RNN-FV pooling outperformed the other pooling methods by a sizable margin. Another interesting observation is that with VGG frame representation, CCA outperformed PCA consistently in all pooling methods. Not shown is the performance obtained when using the activations of the RNN as a feature vector. These results are considerably worse than all pooling methods. Notice that the representation dimension of Mean pooling is 500 (like the features we used), the GMM-FV dimension is $2 \times k \times 500$, where k is the number of clusters in the GMM (this parameter was chosen according to performance on a validation set) and the RNN-FV dimension is 1000.

Table 2 compares our proposed RNN-FV method, combining multiple features together, with recently published methods on both datasets. The combinations were performed using early fusion, i.e., we concatenated the normalized low-dimensional gradients of the models and train multi-class linear SVM on the combined representation. We also tested the combination of our two best models with idt [49] and got state of the art results on both benchmarks. Interestingly, comparable results were obtained even when training the RNN on one dataset and testing on the other, proving that our RNN-FV representation is generic and not dataset specific.

Table 2. comparison to the state of the art on UCF101 and HMDB51. In order to obtain the best performance, we combine, similar to all other contributions, multiple features. We also present a result where idt [49] is combined, similar to all other top results (Multi-skip extends idt). This adds motion based information to our method.

Method	HMDB51	UCF101
idt [49]	57.2	85.9
idt + high-D encodings [53]	61.1	87.9
Two-stream CNN (2 nets) [28]	59.4	88
Multi-skip Feature Stacking [54]	65.4	89.1
C3D (1 net) [14]	–	82.3
C3D (3 nets) [14]	–	85.2
C3D (3 nets) + idt [14]	–	90.4
TDD (2 nets) [5]	63.2	90.3
TDD (2 nets) + idt [5]	65.9	91.5
stacked FV [2]	56.21	–
stacked FV + idt [2]	66.78	–
RNN-FV(C3D + VGG-CCA)	54.33	88.01
RNN-FV(C3D + VGG-CCA) + idt	**67.71**	**94.08**

6.2 Image-Sentence Retrieval

The effectiveness of RNN-FV as sentence representation is evaluated on the bidirectional image and sentence retrieval task. We perform our experiments on three benchmarks: Flickr8K [18], Flickr30K [58], and COCO [48]. The datasets contain 8,000, 30,000, and 123,000 images respectively. Each image is accompanied by 5 sentences describing the image content, collected via crowdsourcing.

The Flickr8k dataset is provided with training, validation, and test splits. For Flickr30K and COCO, no training splits are given, and the splits by [8] are used. There are three tasks in this benchmark: image annotation, in which the goal is to retrieve, given a query image, the five ground truth sentences; image search, in which, given a query sentence, the goal is to retrieve the ground truth image; and sentence similarity, in which the goal is, given a sentence, to retrieve the other four sentences describing the same image. Evaluation is performed using Recall@K, namely the fraction of times that the correct result was ranked within the top K items. The median and mean rank of the first ground truth result are also reported. For the sentence similarity task, only the mean rank is reported.

As mentioned in Sect. 5, we explored RNN-FV based on several RNNs. The first RNN is a generic one: it was trained with the Wikipedia sentences as training data and word2vec as word embedding. In addition, for each of the three datasets, we trained three RNNs with the dataset's training sentences as training data: one with word2vec as word embedding; one with the "CCA word embedding" derived from the semantic vector space of [8], as explained in Sect. 5; and one with

Table 3. Image annotation, image search and sentence similarity results on the Flickr8k, Flickr30k and COCO datasets. Shown are the recall rates at 1, 5, and 10 retrieval results (higher is better). Also shown are the median and mean rank of the first ground truth (lower is better). We compare the results of the previous work to variants of our RNN-FV. The 'wiki' notation indicates that the RNN was trained on Wikipedia and not on the sentences of the specific dataset. Models notated by 'w2v' employ word2vec, while the other models ('cca') use the CCA word embedding (as explained in Sect. 5). 'rvrs' models were trained on reversed sentences. We also report results of combinations: 'cca' and 'reverse' models; 'cca' and the best model (GMM+HGLMM) of [8] ('MM-ENS'); 'cca', 'reverse' and [8]; All RNN-FV models; All RNN-FV models and [8]. The RTP method [57] utilizes additional information that is not accessible to the other methods: manual annotations of bounding boxes in the images, which were collected via crowdsourcing.

		Image Annotation					Image Search					Sentence
		r@1	r@5	r@10	median rank	mean rank	r@1	r@5	r@10	median rank	mean rank	mean rank
Flickr8k Previous	SDT-RNN [33]	6.0	22.7	34.0	23.0	NA	6.6	21.6	31.7	25.0	NA	NA
	DFE [35]	12.6	32.9	44.0	14.0	NA	9.7	29.6	42.5	15.0	NA	NA
	RVP [38]	11.7	34.8	48.6	11.2	NA	11.4	32.0	46.2	11.0	NA	NA
	DVSA [39]	16.5	40.6	54.2	7.6	NA	11.8	32.1	44.7	12.4	NA	NA
	SC-NLM [37]	18.0	40.9	55.0	8.0	NA	12.5	37.0	51.5	10.0	NA	NA
	DCCA [34]	17.9	40.3	51.9	9.0	NA	12.7	31.2	44.1	13.0	NA	NA
	NIC [40]	20.0	NA	61.0	6.0	NA	19.0	NA	64.0	**5.0**	NA	NA
	m-RNN [55]	14.5	37.2	48.5	11.0	NA	11.5	31.0	42.4	15.0	NA	NA
	m-CNN [41]	24.8	53.7	67.1	5.0	NA	20.3	47.6	61.7	**5.0**	NA	NA
	MeanVector [8]	22.6	48.8	61.2	6.0	28.7	19.1	45.3	60.4	7.0	27.0	12.5
	GMM-FV [8]	28.4	57.7	70.1	4.0	20.1	20.6	48.6	64.2	6.0	21.8	10.8
	MM-ENS [8]	31.0	59.3	73.7	4.0	18.4	21.3	50.1	64.8	**5.0**	21.0	10.5
Flickr8K Ours	wiki,w2v	29.3	57.8	70.8	4.0	21.4	19.8	48.5	62.9	6.0	25.2	10.0
	w2v	27.4	57.9	70.5	4.0	22.7	20.4	49.1	63.4	6.0	25.5	10.4
	cca	30.9	60.1	73.1	4.0	19.4	20.7	48.7	63.8	6.0	29.2	11.3
	cca,rvrs	29.1	57.3	71.7	4.0	18.4	20.8	48.5	62.9	6.0	30.2	12.5
	cca + rvrs	30.8	59.8	72.9	4.0	18.2	21.8	49.6	64.4	6.0	27.3	11.2
	cca + [8]	**32.9**	**61.7**	**74.9**	**3.0**	16.8	22.0	51.5	66.5	**5.0**	20.7	9.4
	cca + rvrs + [8]	32.1	60.7	74.8	**3.0**	**16.5**	22.1	51.4	66.5	**5.0**	21.4	9.5
	all rnn-fv models	29.9	60.7	73.4	4.0	17.9	22.4	52.7	67.2	**5.0**	20.9	8.7
	all rnn-fv models + [8]	31.6	61.2	74.3	**3.0**	17.4	**23.2**	**53.3**	**67.8**	**5.0**	**19.4**	**8.5**
Flickr30k Previous	SDT-RNN [33]	9.6	29.8	41.1	16.0	NA	8.9	29.8	41.1	16.0	NA	NA
	DFE [35]	14.2	37.7	51.3	10.0	NA	10.2	30.8	44.2	14.0	NA	NA
	RVP [38]	12.1	27.8	47.8	11.0	NA	12.7	33.1	44.9	12.5	NA	NA
	DVSA [39]	22.2	48.2	61.4	4.8	NA	15.2	37.7	50.5	9.2	NA	NA
	SC-NLM [37]	23.0	50.7	62.9	5.0	NA	16.8	42.0	56.5	8.0	NA	NA
	DCCA [34]	16.7	39.3	52.9	8.0	NA	12.6	31.0	43.0	15.0	NA	NA
	NIC [40]	17.0	NA	56.0	7.0	NA	17.0	NA	57.0	7.0	NA	NA
	LRCN [56]	NA	NA	NA	NA	NA	17.5	40.3	50.8	9.0	NA	NA
	RTP [57](manual annotations)	**37.4**	63.1	74.3	NA	NA	26.0	56.0	69.3	NA	NA	NA
	m-RNN [55]	35.4	63.8	73.7	**3.0**	NA	22.8	50.7	63.1	5.0	NA	NA
	m-CNN [41]	33.6	**64.1**	**74.9**	**3.0**	NA	26.2	**56.3**	69.6	**4.0**	NA	NA
	MeanVector [8]	24.9	52.5	64.4	5.0	27.3	20.5	46.4	59.3	6.8	32.3	16.2
	GMM-FV [8]	33.0	60.8	72.0	**3.0**	19.0	23.9	51.7	64.9	5.0	24.8	15.0
	MM-ENS [8]	35.0	62.1	73.8	**3.0**	17.4	25.1	52.8	66.1	5.0	23.7	14.1
Flickr30k Ours	wiki,w2v	32.9	59.6	72.1	**3.0**	18.5	23.9	52.0	65.2	5.0	26.0	15.2
	w2v	32.0	59.5	71.4	**3.0**	17.2	23.4	51.7	65.2	5.0	24.5	14.1
	cca	33.6	60.5	73.0	**3.0**	15.7	24.5	52.5	66.3	5.0	27.7	16.9
	cca,rvrs	32.8	61.9	72.7	**3.0**	17.4	24.4	51.2	64.6	5.0	28.9	16.1
	cca + rvrs	33.6	62.4	73.4	**3.0**	15.5	25.0	53.6	66.9	5.0	26.2	15.5
	cca + [8]	35.1	63.3	74.2	**3.0**	15.3	26.4	54.9	68.6	**4.0**	21.7	13.4
	cca + rvrs + [8]	35.1	63.5	74.5	**3.0**	**15.0**	26.5	55.2	68.5	**4.0**	22.0	13.5
	all rnn-fv models	34.7	62.7	72.6	**3.0**	15.6	26.2	55.1	69.2	**4.0**	21.2	12.8
	all rnn-fv models + [8]	35.6	62.5	74.2	**3.0**	**15.0**	**27.4**	55.9	**70.0**	**4.0**	**20.0**	**12.2**
COCO Previous	DVSA [39]	38.4	69.9	80.5	**1.0**	NA	27.4	60.2	74.8	**3.0**	NA	NA
	m-RNN [55]	41.0	73.0	83.5	2.0	NA	29.0	42.2	77.0	**3.0**	NA	NA
	m-CNN [41]	**42.8**	73.1	84.1	2.0	NA	**32.6**	**68.6**	**82.8**	**3.0**	NA	NA
	STV [12]	33.8	67.7	82.1	3.0	NA	25.9	60.0	74.6	4.0	NA	NA
	MeanVector [8]	33.2	61.8	75.1	3.0	14.5	24.2	56.4	72.4	4.0	14.7	14.3
	GMM-FV [8]	39.0	67.0	80.3	3.0	11.2	24.2	59.3	76.0	4.0	11.3	12.4
	MM-ENS [8]	39.4	67.9	80.9	2.0	10.4	25.2	59.9	76.7	4.0	11.0	12.9
COCO Ours	wiki,w2v	37.7	70.5	81.0	2.0	9.9	26.6	61.1	76.9	4.0	10.9	11.9
	w2v	39.9	71.5	81.3	2.0	10.5	26.9	61.8	77.4	4.0	11.4	12.1
	cca	40.9	**75.0**	**84.9**	2.0	8.2	30.2	65.0	80.4	**3.0**	11.1	13.2
	cca,rvrs	41.3	71.5	83.7	2.0	**8.1**	28.9	64.5	79.9	**3.0**	11.3	12.6
	cca + rvrs	40.8	73.4	84.1	2.0	8.2	30.4	65.5	80.9	**3.0**	10.7	12.3
	cca + [8]	40.7	72.3	83.5	2.0	9.1	28.1	64.1	79.8	**3.0**	10.2	11.5
	cca + rvrs + [8]	40.2	72.7	84.2	2.0	8.6	29.0	64.8	80.2	**3.0**	10.1	11.5
	all rnn-fv models	40.8	71.9	83.2	2.0	8.9	29.6	64.8	80.5	**3.0**	9.7	10.6
	all rnn-fv models + [8]	41.5	72.0	82.9	2.0	9.0	29.2	64.7	80.4	**3.0**	**9.5**	**10.2**

the CCA word embedding, and with feeding the sentences in reverse order. The RNN is using an LSTM layer of size 100. We did not observe a benefit in using more LSTM units. We used the part of the gradient corresponding to all 30,300 weights of the output layer (including one bias per word-embedding dimension). In the case of the larger COCO dataset, due to the computational burden of the CCA calculation, we used PCA to reduce the gradient dimension from 30,300 to 20,000. PCA was calculated on a random subset of 300,000 sentences (around 50 %) of the training set. We also tried PCA dimension reduction to a lower dimension of 4,096, for all three datasets. We observed no change in performance (Flickr8K) or slightly worse results (Flickr30K and COCO).

Table 3 shows the results of the different RNN-FV variants compared to the baselines and to the current state of the art methods. The baselines, Mean Vector and GMM-FV, appear in the table as previous work of [8]. We also report results of combinations of models. Combining was done by averaging the image-sentence (or sentence-sentence) cosine similarities obtained by each model.

First, we notice the competitive performance of the model trained on Wikipedia sentences, which demonstrates the generalization power of the RNN-FV, being able to perform well on data different than the one which the RNN was trained on. Training using the dataset's sentences only slightly improves results, and not always. Improved results are obtained when using the CCA word embedding instead of word2vec. It is interesting to see the result of the "reverse" model, which is on a par with the other models. It is somewhat complementary to the "left-to-right" model, as the combination of the two yields somewhat improved results. Finally, the combination of RNN-FV with the best model (GMM+HGLMM) of [8] outperforms the current state of the art on Flickr8k, and is competitive on the other datasets.

7 Conclusions

This paper introduces a novel FV representation for sequences that is derived from RNNs. The proposed representation is sensitive to the element ordering in the sequence and provides a richer model than the additive "bag" model typically used for conventional FVs.

The RNN-FV representation surpasses the state-of-the-art results for video action recognition on two challenging datasets. When used for representing sentences, the RNN-FV representation achieves state-of-the-art or competitive results on image annotation and image search tasks. Since the length of the sentences in these tasks is usually short and, therefore, the ordering is less crucial, we believe that using the RNN-FV representation for tasks that use longer text will provide an even larger gap between the conventional FV and the RNN-FV.

A transfer learning result from the image annotation task to the video action recognition task was shown. The conceptual distance between these two tasks makes this result both interesting and surprising. It supports a human development-like way of training, in which visual labeling is learned through natural language, as opposed to, e.g., associating bounding boxes with nouns.

While such training was used in computer vision to learn related image to text tasks, and while recently zero-shot action recognition was shown [59,60], NLP to video action recognition transfer was never shown to be as general as presented here.

Acknowledgments. This research is supported by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI).

References

1. Simonyan, K., Parkhi, O.M., Vedaldi, A., Zisserman, A.: Fisher vector faces in the wild. In: Proceedings BMVC, vol. 1. 7 (2013)
2. Peng, X., Zou, C., Qiao, Y., Peng, Q.: Action recognition with stacked fisher vectors. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 581–595. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10602-1_38
3. Chatfield, K., Lempitsky, V., Vedaldi, A., Zisserman, A.: The devil is in the details: an evaluation of recent feature encoding methods. In: British Machine Vision Conference (2011)
4. Perronnin, F., Liu, Y., Sánchez, J., Poirier, H.: Large-scale image retrieval with compressed fisher vectors. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3384–3391. IEEE (2010)
5. Wang, L., Qiao, Y., Tang, X.: Action recognition with trajectory-pooled deep-convolutional descriptors. arXiv preprint (2015). arXiv:1505.04868
6. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. Adv. Neural Inf. Process. Syst. **28**, 3111–3119 (2013)
7. Lev, G., Klein, B., Wolf, L.: In defense of word embedding for generic text representation. In: Biemann, C., Handschuh, S., Freitas, A., Meziane, F., Métais, E. (eds.) NLDB 2015. LNCS, vol. 9103, pp. 35–50. Springer, Heidelberg (2015). doi:10.1007/978-3-319-19581-0_3
8. Klein, B., Lev, G., Sadeh, G., Wolf, L.: Associating neural word embeddings with deep image representations using fisher vectors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4437–4446 (2015)
9. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, pp. 3104–3112 (2014)
10. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint (2014). arXiv:1409.0473
11. Palangi, H., Deng, L., Shen, Y., Gao, J., He, X., Chen, J., Song, X., Ward, R.: Deep sentence embedding using the long short term memory network: Analysis and application to information retrieval. arXiv preprint (2015). arXiv:1502.06922
12. Kiros, R., Zhu, Y., Salakhutdinov, R., Zemel, R.S., Torralba, A., Urtasun, R., Fidler, S.: Skip-thought vectors. arXiv preprint (2015). arXiv:1506.06726
13. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556 (2014)
14. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. arXiv preprint (2014). arXiv:1412.0767

15. Hotelling, H.: Relations between two sets of variates. Biometrika **17**, 321–377 (1936)
16. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: a large video database for human motion recognition. In: Proceedings IEEE International Conference on Computer Vision (2011)
17. Soomro, K., Zamir, A.R., Shah, M.: UCF101: A dataset of 101 human action classes from videos in the wild. In: CRCV-TR-12-01, November 2012
18. Hodosh, M., Young, P., Hockenmaier, J.: Framing image description as a ranking task: Data, models and evaluation metrics. J. Artif. Intell. Res. (JAIR) **47**, 853–899 (2013)
19. Laptev, I., Marszalek, M., Schmid, C., Rozenfeld, B.: Learning realistic human actions from movies. In: Proceedings IEEE Conference on Computer Vision Pattern Recognition, pp. 1–8 (2008)
20. Wang, H., Klaser, A., Schmid, C., Liu, C.: Action recognition by dense trajectories. In: Proceedings IEEE Conference on Computer Vision Pattern Recognition, pp. 3169–3176 (2011)
21. Kliper-Gross, O., Gurovich, Y., Hassner, T., Wolf, L.: Motion interchange patterns for action recognition in unconstrained videos. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7577, pp. 256–269. Springer, Heidelberg (2012). doi:10.1007/978-3-642-33783-3_19
22. Laptev, I.: On space-time interest points. Int. J. Comput. Vis. **64**(2), 107–123 (2005)
23. Wang, H., Schmid, C.: Action Recognition with improved trajectories. In: International Conference on Computer Vision, October 2013
24. Wang, H., Kläser, A., Schmid, C., Liu, C.L.: Dense trajectories and motion boundary descriptors for action recognition. Int. J. Comput. Vis. **103**(1), 60–79 (2013)
25. Taylor, G.W., Fergus, R., LeCun, Y., Bregler, C.: Convolutional learning of spatio-temporal features. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6316, pp. 140–153. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15567-3_11
26. Ji, S., Xu, W., Yang, M., Yu, K.: 3d convolutional neural networks for human action recognition. Pattern Anal. Mach. Intell. IEEE Trans. **35**(1), 221–231 (2013)
27. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1725–1732. IEEE (2014)
28. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. Adv. Neural Inf. Process. Syst. **25**, 568–576 (2014)
29. Wang, X., Wang, L.M., Qiao, Y.: A comparative study of encoding, pooling and normalization methods for action recognition. In: Lee, K.M., Matsushita, Y., Rehg, J.M., Hu, Z. (eds.) ACCV 2012. LNCS, vol. 7726, pp. 572–585. Springer, Heidelberg (2013). doi:10.1007/978-3-642-37431-9_44
30. Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: Delving deep into convolutional nets. arXiv preprint (2014). arXiv:1405.3531
31. Fernando, B., Gavves, E., Oramas, J.M., Ghodrati, A., Tuytelaars, T.: Modeling video evolution for action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5378–5387 (2015)
32. Nagel, M., Mensink, T., Snoek, C.G.: Event fisher vectors: robust encoding visual diversity of visual streams. Identity **27**(27.2), 22–28 (2015)

33. Socher, R., Le, Q., Manning, C., Ng, A.: Grounded compositional semantics for finding and describing images with sentences. In: NIPS Deep Learning Workshop (2013)
34. Mikolajczyk, F.Y.K.: Deep correlation for matching images and text (2015)
35. Karpathy, A., Joulin, A., Fei-Fei, L.: Deep fragment embeddings for bidirectional image sentence mapping. arXiv preprint (2014). arXiv:1406.5679
36. Mao, J., Xu, W., Yang, Y., Wang, J., Yuille, A.: Deep captioning with multimodal recurrent neural networks (m-rnn). arXiv preprint (2014). arXiv:1412.6632
37. Kiros, R., Salakhutdinov, R., Zemel, R.S.: Unifying visual-semantic embeddings with multimodal neural language models. Trans. Assoc. Comput. Linguist. **2**(10), 351–362 (2015)
38. Chen, X., Zitnick, C.L.: Learning a recurrent visual representation for image caption generation. arXiv preprint (2014). arXiv:1411.5654
39. Karpathy, A., Fei-Fei, L.: Deep visual-semantic alignments for generating image descriptions. Technical report, Computer Science Department, Stanford University (2014)
40. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: A neural image caption generator. arXiv preprint (2014). arXiv:1411.4555
41. Ma, L., Lu, Z., Shang, L., Li, H.: Multimodal convolutional neural networks for matching image and sentence. arXiv preprint (2015). arXiv:1504.06063
42. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
43. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint (2014). arXiv:1412.3555
44. Gordo, A., Gaidon, A., Perronnin, F.: Deep fishing: Gradient features from deep nets. In: Proceedings of the British Machine Vision Conference 2015, BMVC 2015, Swansea, UK, September 7–10, 2015, pp. 111.1–111.12 (2015)
45. Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: Netvlad: Cnn architecture for weakly supervised place recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016
46. Lin, T.Y., RoyChowdhury, A., Maji, S.: Bilinear cnn models for fine-grained visual recognition. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1449–1457 (2015)
47. Perronnin, F., Dance, C.: Fisher kernels on visual vocabularies for image categorization. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2007. pp. 1–8. IEEE (2007)
48. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10602-1_48
49. Wang, H., Schmid, C.: Action recognition with improved trajectories. In: 2013 IEEE International Conference on Computer Vision (ICCV), pp. 3551–3558. IEEE (2013)
50. Vinod, H.: Canonical ridge and econometrics of joint production. J. Econometrics **4**(2), 147–166 (1976)
51. Quasthoff, U., Richter, M., Biemann, C.: Corpus portal for search in monolingual corpora. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation, vol. 17991802 (2006)

52. Perronnin, F., Sánchez, J., Mensink, T.: Improving the fisher kernel for large-scale image classification. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 143–156. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15561-1_11
53. Peng, X., Wang, L., Wang, X., Qiao, Y.: Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice. arXiv preprint (2014). arXiv:1405.4506
54. Lan, Z., Lin, M., Li, X., Hauptmann, A.G., Raj, B.: Beyond gaussian pyramid: Multi-skip feature stacking for action recognition. arXiv preprint (2014). arXiv:1411.6660
55. Mao, J., Xu, W., Yang, Y., Wang, J., Yuille, A.L.: Explain images with multimodal recurrent neural networks. arXiv preprint (2014). arXiv:1410.1090
56. Donahue, J., Hendricks, L.A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. arXiv preprint (2014). arXiv:1411.4389
57. Plummer, B.A., Wang, L., Cervantes, C.M., Caicedo, J.C., Hockenmaier, J., Lazebnik, S.: Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2641–2649 (2015)
58. Hodosh, P., Hockenmaier, J.: From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. Trans. Assoc. Comput. Linguist. **2**, 67–78 (2014)
59. Jain, M., van Gemert, J.C., Mensink, T., Snoek, C.G.M.: Objects2action: classifying and localizing actions without any video example. In: Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, December 2015
60. Xu, X., Hospedales, T.M., Gong, S.: Semantic embedding space for zero-shot action recognition. CoRR abs/1502.01540 (2015)