Nonlinear Channels Aggregation Networks for Deep Action Recognition

Anonymous authors
Paper under double-blind review

Abstract

We introduce the concept of channel aggregation in ConvNet architecture, a novel compact representation of CNN features useful for explicitly modeling the nonlinear channels encoding especially when the new unit is embedded inside of deep architectures for action recognition. The channel aggregation is based on multiple-channels features of ConvNet and aims to be at the spot finding the optical convergence path at fast speed. We name our proposed convolutional architecture "nonlinear channels aggregation networks (NCAN)" and its new layer "nonlinear channels aggregation layer (NCAL)". We theoretically motivate channels aggregation functions and empirically study their effect on convergence speed and classification accuracy. Another contribution in this work is an efficient and effective implementation of the NCAL, speeding it up orders of magnitude. We evaluate its performance on standard benchmarks UCF101 and HMDB51, and experimental results demonstrate that this formulation not only obtains a fast convergence but stronger generalization capability without sacrificing performance.

1 Introduction

With modern learnable representations such as deep convolutional neural networks (CNNs) matured in many image understanding tasks(Krizhevsky et al., 2012), human action recognition has received a significant amount of attentions(Wang & Schmid, 2014; Donahue et al., 2017; Simonyan & Zisserman, 2014; Feichtenhofer et al., 2016; Wang et al., 2015a). Due to the fact that video itself provides an additional temporal clue and that the parameters and the calculations of CNNs grow exponentially, training CNNs with such large-scale parameters in video domain is time-consuming. However, it remains unclear how the effective convergence accelerators could be conducted for the optimal path by formulizing the hand-crafted rules. Since videos consist of still images, training tricks and methods, such as Relu, BN, have been shown to transfer to videos directly. Recent theoretical and empirical works have demonstrated the importance of quickly training deep architectures successfully, and the effective convergence accelerators advanced in the 2D image, such as relu(Glorot et al., 2011) and batch normalization(Ioffe & Szegedy, 2015), have been developed for fast convergence. This is in part inspired by observations of the limited GPU memory and computing power, especially when confronting the large-scale video dataset which may introduce a large majority of parameters. Another pipeline of algorithms focuses on the training optimizer of CNNs, for examples, sgd, momentum, nesterov, adagrad and adadelta. However, training CNNs utilizing the large-scale video datasets is still nontrivial in video task, particularly if one seeks a compact but fast long termporal dynamic representation that can be processed efficiently.

Our current work reconsiders the means of facilitating convergence of ConvNets to increase the understanding of how to embed some hand-crafted rules inside of CNNs for fast convergence in a more thorough fashion. In addition to the accelerators and effective optimizers, we tend to explore a thorough method causing the value of the loss function to descend rapidly. Intuitively, we argue that CNNs will accelerate training process once the complex relationship across convolutional features channels is modeled, explicitly, by the hand-crafted rules. In the existing units 3D convolution implements a linear partial sum of channels (Ji et al., 2013), 3D max-pooling takes the maximum feature by channels and 3D average-pooling make a spatial-channel average of features. Unfortunately, all the 3D units conduct a linear channels aggregation, implicitly and locally. Despite that the implicit linear aggregation has been applied to broad fields, there seems to be less works explicitly taking modeling the complex nonlinear relationship across channels into account. In fact, either one-stream or two-stream algorithms ignore the channel-level encoding. For video recognition task, a very tricky problem is how to train the CNN architectures for the sake of making a lower loss rapidly in the scarcity of videos. We conjecture that there is complex nonlinear relationship among the channels of CNN features. Once this implicit relationship is explicitly modeled, such accomplishment will facilitate converging with faster search to the optimal trajectory.

In this paper, we proposed a nonlinear channels aggregation layer (NCAL), which explicitly models the complex nonlinear relationships across channels. Since a standard CNN provides a whole hierarchy of video representations, the first question worthy exploring is where the NACL should take place. For example, we can aggregate the output of the fully-connected layers of CNN architecture pre-trained on videos. A drawback of such implementation is that the convolutional features channels of CNN itself are still implicitly encoded and are unaware of the lower level channels relationship. The alternative is to model the nonlinear channels aggregation of some intermediate network layer. In this case, the lower layers fail to extract the representative features from video sequences, but the upper layers can reason about the overall dynamics in the video. The former is prone to sacrificing the recognition performance while the latter is thus thought of as the appropriate convolutional features for the compact aggregation.

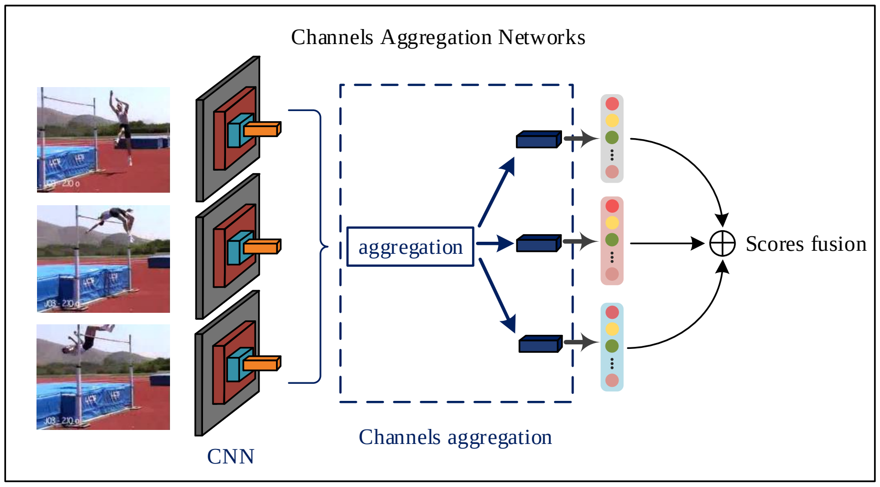

Figure 1: Channels aggregation network: Blue dotted rectangle represents the channels aggregation layer. Along with the channel directions the channels aggregation layer is embodied inside of CNN, and the final feature prediction is obtained by combining information from all the outputs of NCAL.

Here we build our methods on top of the successful Inception V1 architecture. More specifically, three main contributions are provided in this work. **Our first contribution** is to introduce the concept of nonlinear channels aggregation for fast convergence. We also show that, in this manner, it is possible to apply the concept of nonlinear channels aggregation to the intermediate layers of a CNN representation by constructing an efficient nonlinear channels aggregation layer (NCAL).

Here we build our methods on top of the successful Inception V1 architecture. More specifically, three main contributions are provided in this work. **Our first contribution** is to introduce the concept of nonlinear channels aggregation for fast convergence. We also show that, in this manner, it is possible to construct an efficient nonlinear channels aggregation by applying the concept of nonlinear channels aggregation to the intermediate layers of the standard CNN. More importantly, it is explicitly and globally that the nonlinear channels relationship is modeled compared to the traditional local and implicit units.

Our second contribution is to simplify the process of nonlinear channels aggregation layer (NCAL) and make a fast yet accurate implementation of it. Notably, the proposed NCAL can be embodied inside of any standard CNN architectures, and not break the rest components of structures. More broadly, the proposed NCAL is not limited to action recognition, that is, it can be applied to any task with CNNs. Here we introduce it into action recognition, and leave the explorations of it on the other domains in the future.

Our third contribution is to leverage these ideas to construct a novel nonlinear channels aggregation network, perform the training process end-to-end. We show that such nonlinear channels encoding results in a fast decline in the value of the loss function of CNNs while obtains efficient and accurate classification of actions in videos.

The rest of the paper is organized as follows: Section 2 describes the related works, and section 3 represents the principle of the nonlinear channels aggregation networks (NCAN) and the backward propagation of NCAN. This is followed by the experiments in section 4. Finally, we conclude this paper in Section 6.

2 Related works

Owing to the difficulty in training convolutional networks on video dataset, e.g. more parameters and small-scale video sequences, we restrict our works under the premise of action recognition. Previous works related to ours fall into two categories: (1) convolutional networks for action recognition, (2) linear channels aggregation.

Convolutional networks for action recognition. Currently one type of the common practices in the mainstream algorithms enables a stack of consecutive video sequences to train the convolutional architectures, which implicitly captures motion characteristics. Another variants built on this is temporal segment networks (TSN), thus ConvNets are trained leveraging the multiple segments(Bilen et al., 2016; Diba et al., 2016; Wang et al., 2016; 2015b;a). The last practice extends the 2-D convolution to capture the motions with extra temporal dimension. In this case so far the extra temporal dimension introduces exponential parameters. Also, we can make finding that most of effective methods are built on all the basic frameworks presented above. Thus, we perform our methods on these fundamental architectures that most of video classification algorithms have in common.

Linear channels aggregation. In the traditional ConvNets, the implicit relationship across channels can be captured by the 3D convolution, 3D max pooling, 3D average pooling and 3D weighted-average pooling. Compared with 2-D convolution, max pooling, average pooling and weighted-average pooling, these 3D operators conduct the local linear aggregation on the space and channels level. Nevertheless, the guidelines of these simple linear methods fail to model the explicit, global and nonlinear channels relationships. These local linear encoders, in essence, remain unable to completely represent the complex nonlinear channels functions. The proposed nonlinear channels aggregation network, while also emphasizing this principle, is the first framework for end-to-end fast convergence of CNNs by capturing the global channels dependency.

3 Model

In this section, we give detailed descriptions of performing nonlinear channels aggregation. Specifically, we first introduce the basic linear channels encoding concepts in the framework of video tasks and utilize the available layers to conduct the channel-level linear encoding. Then, we introduce the principle of the nonlinear channels aggregation layer (NCAL) simulating the complex channel-level encodings. Finally we describe the fast optimization and backward propagation of NCAL.

3.1 3D convolution channels aggregation

We consider how to leverage current units in the standard CNN to comply the channels relationships. Suppose for the moment that different channels in the convolutional network capture various discriminative features, and for the bold hypothesis that there is a nonlinear relationship between them, stacks of the channels in the subsequent layer will skew CNN in favor of the significant correspondence between these appropriate channels. To achieve the goal, current methods, 3D pooling and 3D convolution can be utilized to achieve it. As mentioned in the introduction, pooling would lead to a decline in feature resolution, which would cause the poor performance of recognition. We thus utilize 3D convolution to implement the linear channels relationship. Given video frames $\mathrm{V} = \{\mathrm{V}_1, V_2, \cdots, V_K\}$ multiple features $\mathrm{S} = \{\mathrm{S}_1, \mathrm{S}_2, \cdots, \mathrm{S}_K\}$ are denoted as the responses of the immediate layer of CNN. Let h, w and c separately represent the height, width and channels, thus $\mathrm{S}_i \in R^{h*w*c}$.

3D convolution. The response at the spatial position (x, y, z) in the j_{th} feature map of the i_{th} layer, defined as $v_{ij}(x, y, z)$, is given by,

$$\begin{aligned} &v_{ij}(x,y,z) \\ &= activate(\sum_m \sum_{i=0}^{I-1} \sum_{j=0}^{J-1} \sum_{c=0}^{C-1} w_{ijm}(i,j,c) * v_{(i-1)m}(x+i, y+j, z+c) + b_{ij}) \end{aligned} \tag{1}$$

where I, J and C are the height, width and channels of the kernel, respectively. *activate* is denoted as the activate functions.

Discuss. Since the numbers of channel is arbitrary, 3D operators, such as 3D max, average pooling and 3D convolution, in the subsequent layers can, implicitly, learn the arbitrary partial correspondence between these local channels. Moreover, pooling operators do not define a local correspondence between channels, but filter the significant features in a space-channels cube, finally leading to a decrease in the resolution of convolutional features. Among these local operators, 3D convolution seems to be a relatively better encoding. In fact, we expect that the 3D encoding between channels cannot be imperceptibly influenced by the spatial encoding, then, encoding spatial and channels relationships simultaneously in the 3D cube, locality and linearity, is not the better model constructing the nonlinear functions.

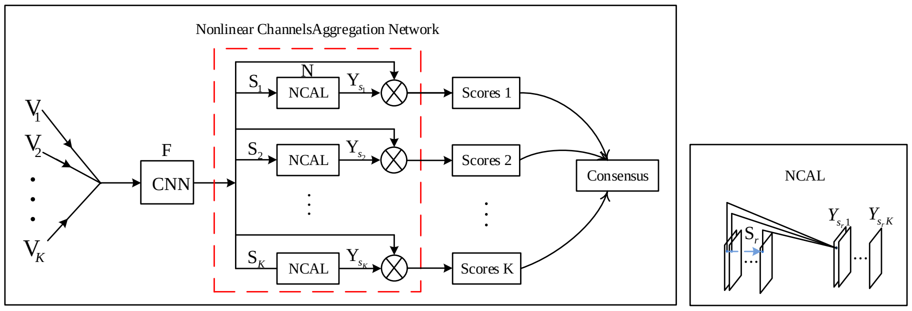

Figure 2: The nonlinear channels aggregation networks for end-to-end global channels aggregation representations. Frame sequences from a video are taken as inputs and fed forward till the end of the nonlinear channels aggregation layer (NCAL). At the NCAL, the video sequences are encoded by channels aggregation operator to produce the same number of channels with the convolutional features, and then is fed to the next layer in the network.

3.2 Nonlinear channels aggregation

In general, pooling is necessary to make network training computationally feasible and, more fundamentally, to allow information aggregation over large areas of the input features. However, when these implementations are utilized to model the complex channel-level relationships, only linear and local channels representations are modeled in an implicit manner. Nevertheless, these temporal pooling and variants tend to result in the reduced resolution, which will achieve coarse classifications. Another downside of the linear channels aggregation is that its locality, inadequate in representing such complex relationship, lacks the global representations across all the channels. To tackle with the problems above, it is desirable to explicitly capture the complex nonlinear relationship encoded in multiple channels. A general design principle of nonlinear channels aggregation layer (NCAL) is that its introduction cannot break the rest of CNN architecture, while keep a unique yet global channel-level encoding for final classification.

Denoting F as the function of CNN, then, S_i is calculated as follows:

$$\mathrm{S}_i = \mathrm{F}(V_i), i = 1, 2, ..., K \tag{2}$$

In particular, let $\mathrm{Y}_{s_r} = \left[\mathrm{Y}_{s_r 0}, \mathrm{Y}_{s_r 1}, ..., \mathrm{Y}_{s_r (C-1)}\right]$ denote the responses of NCAL, one for each of the K video clips. Then, we utilize the channels aggregation equation (3) to encode the channel-level features to produce a compact feature maps group,

$$\begin{aligned} &\mathrm{Y}_{s_r p} = \mathrm{N}_p(S_r) \\ &= \sum_{c=0}^{C-1} \left[\mathrm{H}\left(\mathrm{S}_r(m,n,c) \cdot \cos(\mathrm{G}(cp))\right) + \mathrm{H}\left(\mathrm{S}_r(m,n,c) \cdot \sin(\mathrm{G}(cp))\right)\right] \end{aligned} \tag{3}$$

where G is the linear kernel functions of NCAL, H represents the normalization, m, n represent the spatial location, c denotes the channel location,$r \in [1, K]$, $p \in [0, C-1]$ and$\mathrm{Y}_{s_r} \in R^C$.Instead of many-to-one linear channels aggregation, the NCAL is implement a many-to-many mapping. As shown in Fig.2, each response of NCAL is aggregated by all the features at the same spatial location of S_i producing the global yet compact representation, while no extra parameters are involved. Most importantly, in its input and output, the dimensions remain constant, which means NCAL can be positioned anywhere in the feedforward ConvNet architectures consisting of convolutional, fully connected, pooling and nonlinearity layers. Therefore such implementation enables to keep the rest components in the CNN structure constant with the original, and it is critical to utilize the benefit of transfer learning for other tasks. Next, we multiply every representation Y_{s_i} with individual convolutional feature S_r in the forward propagation pipeline,

$$E_r = \mathrm{Y}_{s_r} * S_r \tag{4}$$

where E_r represents the input features of NACL. Eq. (4) keeps the parameters of layers before NCAL module not being greatly offset by the introduced of nonlinear channels aggregation rules. It can be seen that this implementation, avoiding the vanishing gradient problem, enlarges the gradient of loss with respect to the former layers vanishing gradient problem in the following part.

3.3 Optimization and backward propagation of NCAL

Different from linear channels aggregation unit mapping the same spatial location across the multiple channels to a single feature, the NCAL aims to obtain the same number of global outputs as the inputs. Like the temporal linear encoding networks, one can utilize outer product for capturing the interaction of features with each other at all spatial locations, hence leading to a high-dimensional representation. For this reason, the Tensor Sketch algorithm is utilized to projects this high-dimensional space to a lower-dimensional space. We propose nonlinear channels aggregation layer (NCAL) to fit the complex function from all the channels, and to implement it end-to-end in a fast but efficient manner. Consider convolutional features truncated at a convolutional layer for K segments, these feature maps are matrices $\mathrm{S} = \{\mathrm{S}_1, \mathrm{S}_2, \cdots, \mathrm{S}_K\}$ A channels aggregation function $\mathrm{N} : \mathrm{S}_1, \mathrm{S}_2, \cdots, \mathrm{S}_K \rightarrow \mathrm{Y}_{s_1}, \mathrm{Y}_{s_2}, \cdots, \mathrm{Y}_{s_K}$, aggregates c channels in each segment to output multiple encoded representations $\mathrm{Y}_{s_1}, \mathrm{Y}_{s_2}, \cdots, \mathrm{Y}_{s_K}$, and this unit can be applied to the output of intermediate convolutional layers. In order to embed NCAL inside of CNNs as an intermediate layer, it is critical to compute the fast forward propagation, the gradients for the back-propagation step and to allow back-prop training through it.

First, we implement a skilled and equivalent forward propagation. Based on this, the specific gradient of NCAL is derived for the back-propagation. We divide eq. (4) into two parts,

$$\mathrm{Y}_{s_r p} = \mathrm{Y}^1_{s_r p} + \mathrm{Y}^2_{s_r p} \tag{5}$$

$$\mathrm{Y}^1_{s_r p} = \sum_{c=0}^{C-1} \mathrm{S}_r(m,n,c) \cdot \cos(\mathrm{G}(cp)) \tag{6}$$

$$\mathrm{Y}^2_{s_r p} = \sum_{c=0}^{C-1} \mathrm{S}_r(m, n, c) \cdot \sin(\mathrm{G}(cp)) \tag{7}$$

where the normalization process H is removed for simplifying the process. We can find that, as shown in eq. (6) and (7), such accumulation across all the channels for all the segments suffer enormous computation complexity equivalent to the outer product operators. To tackle the optimization difficulty, we redefine eq. (6) and (7) to the novel matrix form,

$$\begin{aligned} &\mathrm{Y}^L_{s_r} = \{\mathrm{Y}^L_{s_r p}\}_{p\in[0,C-1]} \\ &= W^L_{s_r} \cdot S_r(m, n, :), L = \{1, 2\} \end{aligned} \tag{8}$$

where $W^L_{s_r}$ represents the corresponding transformation matrix. More specifically, the corresponding matrix form can be formulized as,

$$W^2_{s_r} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \cos(\mathrm{G}(1)) & \cos(\mathrm{G}(2)) & \cdots & \cos(\mathrm{G}(\mathrm{C}-1)) \\ 1 & \cos(\mathrm{G}(2)) & \cos(\mathrm{G}(4)) & \cdots & \cos(\mathrm{G}(2(\mathrm{C}-1))) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \cos(\mathrm{G}((\mathrm{C}-1))) & \cos(\mathrm{G}(2(\mathrm{C}-1))) & \cdots & \cos(\mathrm{G}((\mathrm{C}-1)(\mathrm{C}-1))) \end{bmatrix} \tag{9}$$

$$W^2_{s_r} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \sin(\mathrm{G}(1)) & \sin(\mathrm{G}(2)) & \cdots & \sin(\mathrm{G}(\mathrm{C}-1)) \\ 1 & \sin(\mathrm{G}(2)) & \sin(\mathrm{G}(4)) & \cdots & \sin(\mathrm{G}(2(\mathrm{C}-1))) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \sin(\mathrm{G}((\mathrm{C}-1))) & \sin(\mathrm{G}(2(\mathrm{C}-1))) & \cdots & \sin(\mathrm{G}((\mathrm{C}-1)(\mathrm{C}-1))) \end{bmatrix} \tag{10}$$

All the steps we have done so far, whether $\mathrm{Y}^1_{s_r}$ or $\mathrm{Y}^2_{s_r}$, transform time-consuming loop traversal problem to matrix multiplication. As noted above, with the advantage that the input and output have the same dimensions NCALs can be applied at any point in the convolutional networks. After constructing the fast forward propagation, we analyze the formulization of a conventional derivative solution, when NCAL is embedded in CNN architectures as a single layer. In particular, with $S^{l-1}_r(0), S^{l-1}_r(1), ..., S^{l-1}_r(C-1)$ denoted as the feature maps of S_r at the l−1 layers of the architecture, Y_{s_r} can be mapped as:

$$\mathrm{Y}_{s_r} = \mathrm{N}(S^{l-1}_r(0), S^{l-1}_r(1), ..., S^{l-1}_r(C-1)) \tag{11}$$

After this, the back-propagation step can be conducted by the gradient,

$$\frac{\partial \mathrm{Y}_{s_r}}{\partial S^{l-1}_r} = (W^1_{s_r} + W^2_{s_r})^T \tag{12}$$

Discuss. Next, we discuss the difference of nonlinear channels aggregation layer compared with the traditional outer product. The benefit of the original outer product is that the resulting feature captures multiplicative interactions at corresponding spatial locations. The main drawback of this feature is its high dimensionality. Once the resulting features are followed by a fully-connected layer, many more parameters will be involved in the CNN, prone to network over-fitting. So far we can find that the NCAL, as well as the outer product, captures the global information representing the whole channels range and the NCAL is optimized by a fast yet compact implementation of matrix. Not only that, but each value in the weighting kernel in the NCAL is a single mapping for channels while the corresponding value in the outer product is many-to-one mapping, that is, outer product between the whole channels range views each channel equally important. In our proposed methods, all the responses of NCAL are obtained, only once, by the multiplication of transformation matrix with the convolutional feature maps of an intermediate layer which the outer-product implementation that every response of NACL is aggregated by every channel of one intermediate layer is transformed to matrix multiplication.

4 Experiments

We first give a detailed description for the experimental setup used in the paper. Then we evaluate and compare the performances of NCAN architectures in the following sections. Finally the change curse of loss with iteration is depicted for further analyzing its effect.

4.1 Datasets and Experimental setup

UCF101. UCF101 is an action recognition dataset, collected from YouTube, with 101 video categories (Soomro et al., 2012). The videos in 101 action categories are grouped into 25 units, each unit containing 4-7 videos of an action. With large variations in camera motion, object appearance and pose, viewpoint, clutter background, etc, it is a challenging dataset.

HMDB51. HMDB51 is a large human motion dataset, containing 6849 clips divided into 51 action categories (Kuehne et al., 2011). Due to that HMDB 51 is extracted from commercial movies as well as YouTube, it can represent a sufficient reflect of light conditions, situations and surroundings, which are close to the realistic video.

Experimental setup. In view of the efficiency and accuracy, our architectures are built on the Inception V1, and pre-trained on ImageNet dataset. With the pre-trained model initializing our architectures all the layers are fine-tuned with the learning rate set to 10^{-2}, decrease it to 10^{-3} after 12 epochs and stop the training process after 20 epochs. We position the NCAL behind the different convolutional layer, thus, various nonlinear aggregation networks with NCAL in different location are constructed to explore where the NCAL should be placed.

4.2 Accuracy comparisons of CNNs followed by the NCAL in different convolutional layers

In this section, we sample a single image from a video for training and evaluate the performances of CNN with NACL in various convolutional layers. Table I reports the experimental results of CNNs with NCAL in various location on the split 1 of UCF101. We can see that the NCAL following the last convolutional layer outperforms the reproduction of the original architecture, and the NCAL performs well when placed behind the latter convolutional layer. As shown in Table I, the more distinctive the convolutional features are, the higher, on accuracy of recognition, is the effect of CNN with the NCAL. Based on this, we stack multiple frames to train CNNs with NCAL, embedded inside of the final convolutional layer, and report the performances on the three splits of UCF101 and HMDB51.

Table 1: Experimental results of NCAL on various locations on the split1 of UCF101.

Dataset	None	Data	Conv1	Conv2	Conv3c	Conv4e	Conv5a	Conv5b	Conv5a+Conv5b
UCF101	79.5%	78.7%	75.5%	67.2%	75.6%	79.2%	78.94%	79.8%	80.0%

Table 2: Experimental results of channels aggregation on the three splits of UCF101 and HMDB51.

Dataset	standard CNN	3D convolution	NCAN
UCF101	82.7%	82.0%	83.4%
HMDB51	49.7%	45.1%	50.0%

Among the NCANs the CNN with multiple NCALs turn out to be most competitive. In practice, on accuracy of recognition, our NCANs make marginal improvements of 0.3%-0.5% (single frame in table I) and 0.3%-0.7% (stacked frames in table II). This can be ascribed to that the channel-level aggregation maps the convolutional maps to another distinctive feature space, in which each feature map is encoded by the hand-crafted rules of feature enhancement. Another intuitive conjecture is that the NCAL captures the global nonlinear property across the channels, resulting in the improvement of accuracy. However, the purpose of this paper is not to boost the performance of action recognition but to facilitate convergence without scarifying classification accuracy. As another interesting noting, no extra parameters are updated in the NACL, in comparison to CNN which purely consisting of convolutional layer, pooling layer and fully-connected layers and so on. Thus, our models are constructed simple, and computationally efficient. Moreover, there is no need rebuilding the CNN models when introducing the NCAL.

4.3 Explorations of convergence between NCANs with the standard CNN

In the case seen in the training process of CNN, the network will almost lead to a multi-fold increase in training time once trained by stacking the still images. Therefore, it is critical to arrive at the same loss within fewer iterations. we evaluate in Figure 3 the convergence of NCANs, and compare the standard CNNs with and without NCALs in terms of convergence. Experimental results have demonstrated that, on the split 1 of UCF101, the NCANs make marginal performance in comparison to the standard CNNs. More importantly, we observe that NCANs converge faster than the original CNNs.

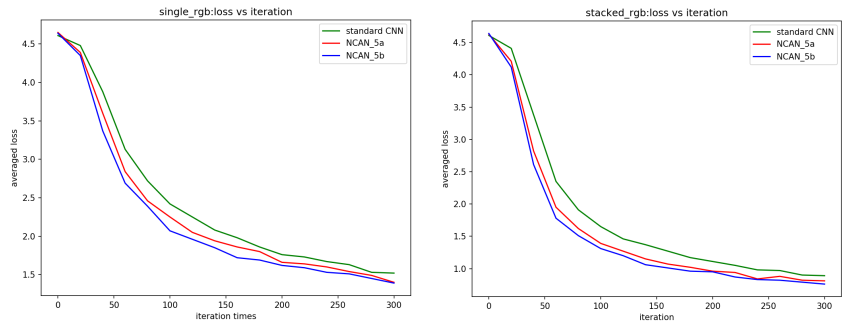

Figure 3: Loss fluctuation curve of iteration in the training process. NACN_5a represents the standard CNN with NCAL placed behind in the 5a convolutional layer. Likewise, NCAN_5b is constructed by positioning NCAL behind the 5b convolutional layer.

Among the 6 splits of UCF101 and HMDB51, the proposed NACNs achieve superior performance of convergence than the standard CNN which is shown in Figure 4. We conclude that nonlinear channels aggregation is a high-quality encoding of channels relationship, while to be a global fitting to complex channels functions as it aggregates all the channels features to a new dimension space.

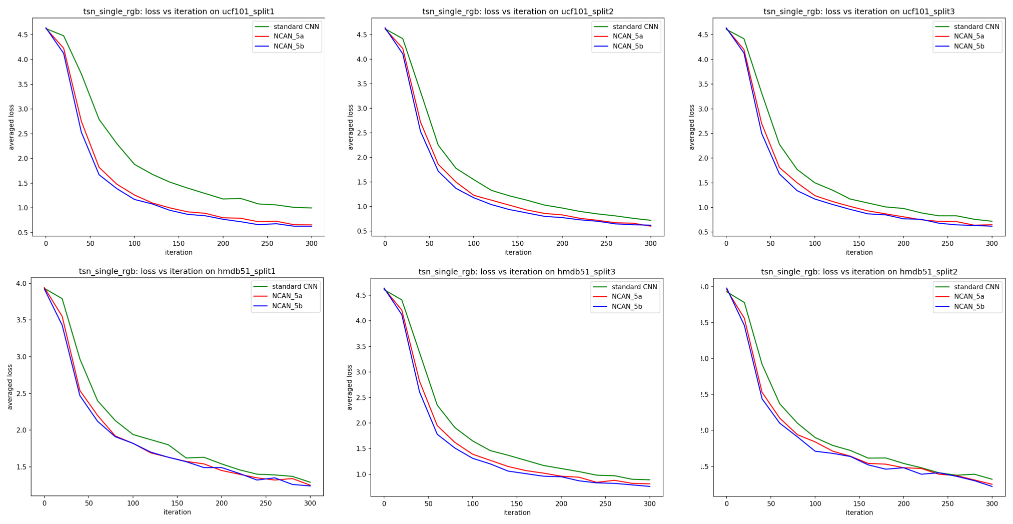

Figure 4: The loss of NCAN with its iterations. The first line represent the convergence comparisons of NACN with the standard CNN on the three splits of UCF101, and the comparisons of convergence between NACN and the standard CNN is shown in the second one.

From a modeling perspective, we are interested in answering the following questions: what hand-crafted rules are best at taking advantage of global channels relationships in standard CNN architecture? How do such hand-crafted rules influence the convergence of a CNN? We limit the hand-crafted rule under the constraint of channels enhancement, and examine these questions empirically by taking a nonlinear channels aggregation to model the complex and nonlinear relationships across the channels. From a practical standpoint, currently there are few methods that conduct the global yet explicit channels dependency because exploring global channels relationship are significantly more difficult to ascertain and model. That is, most of channels relationships are local, vague and implicit. For the explicit and global channels aggregation, we propose nonlinear channels aggregation operations in the section 3.1. To keep the rest parts of the convolutional structure constant with the original CNNs, both linear and nonlinear channels aggregations use the convolutional features as input and the same number of channels as its output. Such setting is significant to extend the channels aggregation to other domains in the future.

4.4 Why NCAL can facilitate network convergence.

In addition to the convergence comparison of NCAN with the standard CNN, a theoretical analysis why NCAL can facilitate training network should be considered. We therefore perform a qualitative analysis in the forward and backward propagations. As explained in the previous section, the NCAL can be conducted as a nonlinear combination of data points along the channels and coefficients can be computed by using the eq. (5). In our work, if the normalization H is ignored, these parameters can also be defined as a one-to-one mapping in Equ.(12).Each item in eq. (12) is greater than 1 when $\theta \in [0, \frac{\pi}{2}]$, as shown in Fig. 5.

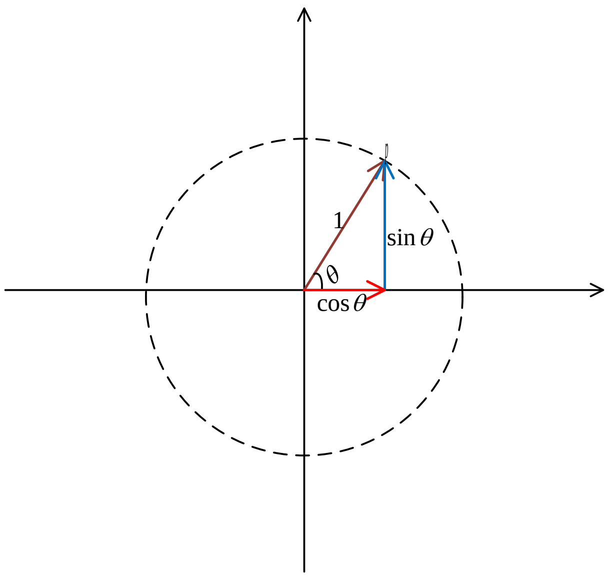

Figure 5: Given arbitrary angle θ and assuming that the end edge of any angle intersects with the unit circle at the point p , a directed triangle then is construct. Naturally, $\sin\theta$ and $\cos\theta$ represent two sides of this triangle,$\sin\theta + \cos\theta \geq 1$,$\theta \in \left[0, \frac{\pi}{2}\right]$

In practice, owing to the periodicity of sinusoidal and cosine functions, the mapping process may result in the mistake that the feature in different channels is mapped to the same value. This mapping mistake will make CNN confused and lose the ability of recognizing the features with characteristics with periodic relations, which reduces the accuracy of recognition. To tackle this problem, we propose the kernel function of NCAL G rescales θ to a single-valued interval $\theta \in \left(0, \frac{\pi}{2}\right)$, such that each feature in the convolutional is enhanced and aggregated by distinctive transformation items. For the back-propagation, the gradients have been derived in eq. (12), and fortunately is a symmetric positive definite matrix. It is interesting to note that the proposed nonlinear channels aggregation layer not only captures the global channels relationship but enlarges the gradient in the back-propagation, reducing the risk of vanishing gradients.

We, furthermore, perform an analysis of nonlinear channels aggregation layer (NCAL). In general, representations on later convolutional layers tend to be somewhat local, where channels correspond to specific, natural parts instead of being dimensions in a completely distributed coding. That said, not all channels correspond to natural parts, resulting in a possible and different decomposition of discriminative features than humans might expect. We can even assume that partial channels are useless or interfering, inevitably, this will become a key point that leads to retarded network convergence. In particular, NCANs perform well on CNNs with methods facilitating convergence, for instance, BN, dropout and optimization strategies, which indicate that the NCAL is complementary to the existing training tricks.

5 Conclusions

We present nonlinear channels aggregation, a powerful and new, yet simple concept in the context of deep learning that captures the global channels relationship. We introduce a novel nonlinear channels aggregation layer (NCAL) and make a fast yet accurate implementation of NCAL, which allows us to embed the principle of complex channels encoding to the mainstream CNN architectures and back-propagate the gradients through NCALs. Experiments on video sequences demonstrate the effective power of nonlinear channels aggregation on facilitating training CNNs.

In this paper we fit the complex channels relationships by capturing the global channels aggregation. Still, there seems to be some possible research directions that can be further expanded, modeling the nonlinear functions across channels. In the future it is beneficial to explore multiple-scale channel-levels by pyramid coding across channels. In sublimation, we can embed any hand-crafted rules, channels aggregation in the mainstream architectures, to making CNN working as we expect.

References

Hakan Bilen, Basura Fernando, Efstratios Gavves, Andrea Vedaldi, and Stephen Gould. *Dynamic Image Networks for Action Recognition.* 2016.

Ali Diba, Vivek Sharma, and Luc Van Gool. Deep temporal linear encoding networks. 2016.

Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. *IEEE Transactions on Pattern Analysis & Machine Intelligence*,39(4):–691, 2017.

Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. *Convolutional Two-Stream Network Fusion for Video Action Recognition.* 2016.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, pp. 315–323, 2011.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. pp. 448–456, 2015.

Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence*,35 (1):–231, 2013.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. *ImageNet classification with deep convolutional neural networks.* 2012.

H Kuehne, H Jhuang, E Garrote, T Poggio, and T Serre. *HMDB: A Large Video Database for Human Motion Recognition.* 2011.

Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. 1(4):568–576, 2014.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *Computer Science*, 2012.

Heng Wang and Cordelia Schmid. *Action Recognition with Improved Trajectories.* 2014.

Limin Wang, Yu Qiao, and Xiaoou Tang. *Action recognition with trajectory-pooled deep-convolutional descriptors.* 2015a.

Limin Wang, Yuanjun Xiong, Zhe Wang, and Yu Qiao. Towards good practices for very deep two-stream convnets. *Computer Science*, 2015b.

Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. *Temporal Segment Networks: Towards Good Practices for Deep Action Recognition.* 2016.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under grant No. 61501357 and Basic science research project of Shaanxi Province under grant No. 2016JQ6080.