

MIRU2018若手プログラム グループサーベイ法

片岡 裕雄, *Ph.D.*

<http://www.hirokatsukataoka.net/>

片岡 裕雄（かたおか ひろかつ）

産業技術総合研究所 CV研究グループ/AL連携研究室/AIセンター 研究員
東京電機大学客員研究員



Hirokatsu KATAOKA, *Ph.D.*

片岡 裕雄

Research Scientist, AIST, Japan.

2014年慶應義塾大学大学院理工学研究科修了，博士（工学）。2013，2014年 ミュンヘン工科大学 Visiting Scientist，2014年東京大学博士研究員，2015年産総研特別研究員。2016年4月より現職。画像認識，動画解析，人物行動解析に興味を持つ。cvpaper.challenge主宰。2011年ViEW小田原賞，2013年電気学会誌論文奨励賞，2014年藤原賞，2016年ECCV WS Brave New Idea Award。

mypage: <http://www.hirokatsukataoka.net/>

cvpaper.challenge: http://hirokatsukataoka.net/project/cc/index_cvpaperchallenge.html

片岡の主宰するcvpaper.challenge

論文読破・まとめ・発想・議論・実装・論文執筆・（社会実装）に至るまで取り組むCVの今を映す挑戦

- 人員：産総研/筑波大/電機大/慶應大/早大/東大による約30名
- BraveNewなアイデアをトップ国際会議*に投稿

* Google Scholar Top-20にリストアップされている国際会議や論文誌

年間1,000本以上、累計2,500本以上のスライドを作成
本取り組みの結果10本以上の論文（含CVPRx2, ICRA, BMVC, ICPRx2, CVPRWx6, ECCVWx2, ICCVW）が採択
8件の招待講演、3件の国内外での受賞

CV分野の今を映すcvpaper.challengeの取り組み

IS1：6月13日(水) コアタイム：14:15-16:00（インタラクティブセッション会場）



片岡 裕雄 氏（産業技術総合研究所）

概要

コンピュータビジョン分野の今を映し、創り出す挑戦です。論文読破・まとめ・アイデア考案・議論・実装・論文執筆（・社会実装）に至るまで広く取り組み、あらゆる知識を共有しています。現在までにCVPR論文を完全読破するなど年間1,000本以上、累計では2,500本の論文をまとめています。最近では学生を中心に論文投稿してトップ国際会議のCVPR/ICRAなどにも採択されるまでになりました。2018年は(i) トップ国際会議/学術論文誌に20本以上投稿する、(ii) CVPR2018論文を完全読破する、を目標に活動しています。今回のインタラクティブ発表ではサーベイやアイデア発想法、組織化した研究法と、その成果などをご紹介します。日本のCVを強くするためにも、是非一緒にディスカッションしましょう！

略歴

産業技術総合研究所コンピュータビジョン研究グループ研究員。2014年慶應義塾大学大学院理工学研究科修了、博士（工学）。2013、2014年ミュンヘン工科大学Visiting Scientist、2014年東京大学博士研究員、2015年産総研特別研究員。画像認識、動画解析、人物行動解析に従事。cvpaper.challenge主宰。2011年VIEW小田原賞、2013年電気学会誌論文奨励賞、2014年藤原賞、2016年ECCV WS Brave New Idea Award。



HP, Twitter, SlideShareもご覧ください

HP: http://hirokatsukataoka.net/project/cc/index_cvpaperchallenge.html

Twitter: @CVpaperChalleng

SlideShare: @cvpaperchallenge

サーベイチュートリアル

- 前半: 米谷
 - 特にサーベイを論文化するうえでどういう点に留意すべきか
- 後半: 片岡
 - グループでサーベイをするうえでどういう点に留意すべきか
- 本講演の様子およびスライドは後日若手Pウェブサイト上で公開します

サーベイについて

サーベイとは？

ひとことでいうと、その分野の動向を把握すること

- 現在どんな技術が流行っている？
- どういう歴史を辿ってきた？
- 自分のやっていることに最も近いものは？

II. RELATED WORK

A. Traffic data and approaches to its representation

Several practical databases for pedestrian detection, such as the French Institute for Research in Computer Science and Automation (INRIA) Dataset [3], Caltech [4], and the KITTI Vision Benchmark Suite for self-driving cars [2]) have been proposed in the past decade. The information contained in the KITTI database, which has been used to set meaningful vision problems for self-driving cars [2] as well as problems related to stereo vision, optical flow, visual odometry, semantic segmentation, two- and three-dimensional (2D/3D) object detection, and 2D/3D tracking, has proven especially useful.

In 2015, these problems were updated for stereo and optical flow [5]. Thanks to the development of sophisticated approaches, such as fully convolutional networks (FCN) [6] and region-based convolutional neural networks (R-CNN) [7], there has been improved performance of solving these problems using the KITTI benchmark dataset. In addition, a manner of geometry allows us to improve the rate of object detection [8] and optical flow [9] not only in stereo [10]. As for semantic segmentation, we can now obtain knowledge about dense connections and use this information with graphical models [11], [12].

Unfortunately, none of these datasets contain scenes of near-miss incidents in which pedestrians, cyclists, or other vehicles must be avoided. Thus, there is an urgent need for a collection of incident scenes that can be used to train self-driving cars on how to safely navigate such dangerous situations.

論文にもRelated workを書くこと多し

なぜ、サーベイをするのか？

トレンドの把握

- 知識がないと既存研究の劣化版を作りかねない
- トレンドを知らないと(天才でない限り)最先端の研究を生み出すことは難しい

自身の研究の立ち位置を確認

- 何が違う？なぜやる？どこが良いのか？という哲学

究極的には次のトレンドを作るため（ここ重要）

- 分野の方向性を自ら定める
- より良く、正しい方向へ導く

どのように，サーベイをするのか？

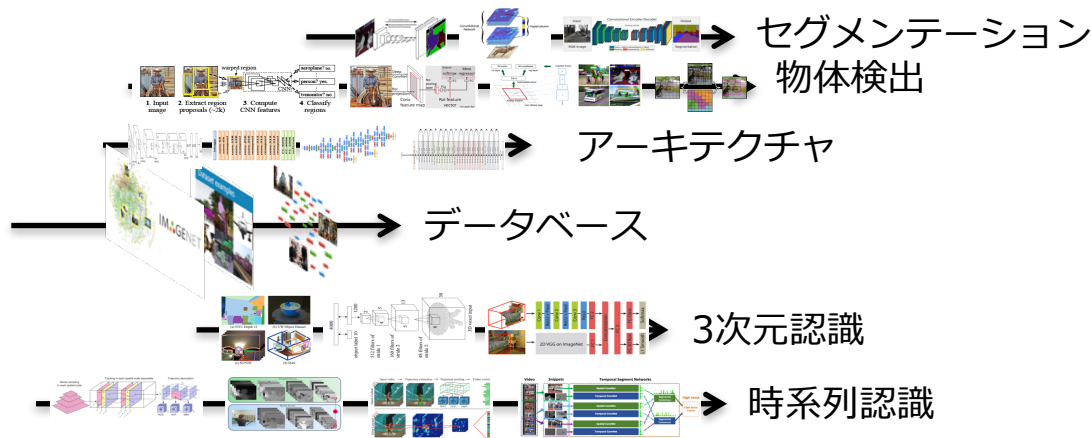
「網羅的に調べる」ことで分野の状況を高精度に概観

- サーベイの質や量が研究テーマ考案・選定に直結
- （研究室単位で）1,000⁺本/年は論文を読む力をつけよう（速読/精読が重要）

「歴史/トレンド」のストリームを多数把握

- 点より線/面で技術を捉える（他分野からも学ぶ）
- 次に何をすれば良いか/その技術の本質や行間が，肌感覚でわかる

「解かれた問題」が犇めく空間を埋めつつ「解かれていない問題を探索」



どれくらいサーベイをしてるの？

個人/グループとしてサーベイを推進

－ 「個人」で達成

- 2015年度 615本, 2016年度 400+本
- うち速読900本, 精読100本くらい？

－ 「グループ」で達成

- 2015年, CVPR 2015 完全読破 (約10名)
- 2016年, 1,000本読破達成 (約20名)
- 2018年, CVPR 2018 完全読破チャレンジ実行中 (約30名)

<http://hirokatsukataoka.net/project/cc/cvpr2018survey.html>



意識的にサーベイして何が変わった？

2015年以前：個人プレー

- 論文調査：自分の狭い分野のみ
- 研究：従来法の単純な改善

組織的にサーベイしてから...

- 論文調査：網羅的かつ流れを把握
- 研究：物事の本質に迫るような問いを意識
- サーベイ/テーマ設定/実験/論文執筆に至るまで「質・効率を高める努力」を徹底
 - 探索は終わらない！

優れた問いを見つけよう！

思いつき（単純拡張）研究から離脱しよう

- To invent, you need a good imagination and **a pile of junk**
 - 良い発想と**ガラクタの山（膨大な知識量）**が必要だ
- 知識を詰め込む/整理することで視野が繋がる
 - 各個人で頑張る（量増加）, 集団で整理して体系化（質向上）
 - 個人でモチベーションを保ち, チーム力を発揮できる体制を整える

個人とグループの知識獲得がキモ

個人のサーベイ

グループのサーベイ

個人のサーベイ
(分割)

グループのサーベイ
(統合)

「分割と統合」の組み合わせが重要

個人サーベイ

個人サーベイで意識すること

速読と精読の組み合わせ

－ 速読（50+本/月）

- とにかく「広く浅く」
- 研究テーマやアイデア考案の時に行うサーベイ法

－ 精読（10-本/月）

- 実装レベルで「狭く深く」
- 具体的なテーマが決まっている際に

尺度を変えた読み方の統合で研究の効率化

論文を「読む」のではなく「読む」論文を「読む」

Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet?

Kensho Hara, Hirokatsu Kataoka, Yuta Satoh
National Institute of Advanced Industrial Science and Technology (AIST)
Tsukuba, Ibaraki, Japan
(hara.kensho, kataoka.hirokatsu, yuta.satoh@aist.go.jp)

Abstract

The purpose of this study is to determine whether current video datasets have sufficient data for training very deep convolutional neural networks (CNNs) with spatiotemporal three-dimensional (3D) networks. Recently, the performance of 3D CNNs in the field of action recognition has improved significantly. However, in data, conventional research has only explored relatively shallow 2D architectures. We examine the architecture of various 3D CNNs from relatively shallow to very deep ones on current video datasets. Based on the results of these experiments, the following conclusions could be obtained: (1) ResNet-18 training resulted in significant overfitting for UCF-101, HMDB-51, and Kinetics-400, but not for Kinetics-610. (2) The Kinetics dataset has sufficient data for training of deep 3D CNNs, and enables training of up to 152 ResNet layers, successfully similar to 2D ResNet on ImageNet. ResNet-10 achieved 73.6% average accuracy on the Kinetics test set. (3) ResNet-10 pre-trained on ImageNet outperforms other deep 3D architectures, and outperforms ResNet-10 on ImageNet by 2.9% and 72.5% on UCF-101 and HMDB-51, respectively. (4) The use of 3D CNNs trained on Kinetics has produced significant performance in action recognition. We believe that using deep 3D CNNs together with Kinetics will retrace the history of 2D CNNs and ImageNet, and stimulate advances in the field of action recognition. We provide pretrained models used in this study and publicly available.

1. Introduction

The use of large-scale datasets is extremely important when using deep convolutional neural networks (CNNs) for various machine learning tasks, and the use of CNNs in the field of computer vision has expanded significantly in recent years. ImageNet [1], which includes more

https://github.com/kenshohara/3D-ResNets-PyTorch

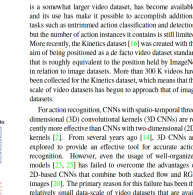


Figure 2: Average accuracy of 3D ResNet over 1 and 3 epochs. The graph shows accuracy vs. number of epochs for ResNet-10, ResNet-18, and ResNet-34. ResNet-10 reaches ~0.74, ResNet-18 ~0.73, and ResNet-34 ~0.72.

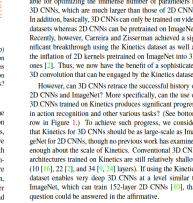


Figure 3: ResNet accuracy to compare video to images (top) and video between. The graph shows accuracy vs. number of layers for ResNet-10, ResNet-18, and ResNet-34. ResNet-10 reaches ~0.74, ResNet-18 ~0.73, and ResNet-34 ~0.72.

2. Related Work

The HMDB-51 [17] and UCF-101 [18] datasets are currently the most used in the field of action recognition. These datasets gained significant popularity in the early years of the field, and are still used in popular benchmarks. However, a recent consensus has emerged that indicates that they are simply not large enough for training deep CNNs from scratch.

A couple of years after the aforementioned datasets were introduced, larger video datasets were produced. These include ActivityNet [3], which contains 80 hours of videos,

Table 2: Accuracies on the Kinetics test set. Average is averaged across one ResNet and Top-5.

Model	Top-1	Top-5	Average
UCF-101	54.2	78.2	66.1
ResNet-34	60.1	81.9	71.0
ResNet-50	62.3	83.3	72.7
ResNet-101	62.8	83.9	73.3
ResNet-152	63.0	84.4	73.7
Two-stream [20]	63.4	84.4	73.7

comparisons with other architectures are shown in Table 1. Here, it can be seen that the accuracies of pre-trained ResNet-20 are slightly less than when compared with the standard ResNet-20 through the use of Kinetics. However, the pre-activation losses overfitting and ImageNet 2D ResNet-20 on ImageNet [11]. We can also see that the 3D ResNet achieved higher accuracies when compared with the ResNet-152, which is similar to the results on ImageNet.

Table 3: Accuracies on the Kinetics test set. Average is averaged across one ResNet and Top-5. Here, we refer the results of RGB and Two-stream 3D trained from scratch [3] for fair comparison.

Model	Top-1	Top-5	Average
UCF-101	54.2	78.2	66.1
ResNet-34	60.1	81.9	71.0
ResNet-50	62.3	83.3	72.7
ResNet-101	62.8	83.9	73.3
ResNet-152	63.0	84.4	73.7
Two-stream [20]	63.4	84.4	73.7

comparisons with other architectures are shown in Table 1. Here, it can be seen that the accuracies of pre-trained ResNet-20 are slightly less than when compared with the standard ResNet-20 through the use of Kinetics. However, the pre-activation losses overfitting and ImageNet 2D ResNet-20 on ImageNet [11]. We can also see that the 3D ResNet achieved higher accuracies when compared with the ResNet-152, which is similar to the results on ImageNet.

Table 4: Accuracies on the Kinetics test set. Average is averaged across one ResNet and Top-5. Here, we refer the results of RGB and Two-stream 3D trained from scratch [3] for fair comparison.

Model	Top-1	Top-5	Average
UCF-101	54.2	78.2	66.1
ResNet-34	60.1	81.9	71.0
ResNet-50	62.3	83.3	72.7
ResNet-101	62.8	83.9	73.3
ResNet-152	63.0	84.4	73.7
Two-stream [20]	63.4	84.4	73.7

comparisons with other architectures are shown in Table 1. Here, it can be seen that the accuracies of pre-trained ResNet-20 are slightly less than when compared with the standard ResNet-20 through the use of Kinetics. However, the pre-activation losses overfitting and ImageNet 2D ResNet-20 on ImageNet [11]. We can also see that the 3D ResNet achieved higher accuracies when compared with the ResNet-152, which is similar to the results on ImageNet.

including 230,000 action instances. ActivityNet also provides some additional tasks, with a multimodal classification and detection, but the number of action instances is still on the order of tens of thousands. This year (2017), we introduced a coarse-to-fine pre-trained model [14] based on Kinetics dataset [14]. The Kinetics dataset includes more than 300,000 training videos covering 400 classes. In order to determine whether it can train deep 3D CNNs, we performed a number of experiments using these recent datasets, as well as the UCF-101 and HMDB-51 datasets.

Other large datasets, such as Sports-1M [1] and YouTube-8M [1] have been proposed. Although these datasets are larger than Kinetics, their annotations are slightly sparser and only video-level labels have been provided. In other words, they include frames that do not relate to target activities. Such noise and the presence of unrelated frames have the potential to confuse these models from modeling and learning. In addition, with the size in excess of 10 TB, their scales are simply too large to allow them to be utilized easily. Because of this, we decided to refrain from discussing these datasets in this study.

2. Action Recognition Architectures

One of the popular approaches to CNN-based action recognition is the use of two-stream CNNs with 2D convolutional kernels. In their study, Simonyan et al. proposed a method to use RGB and optical-flow frame flows to represent and reason information, respectively [21], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance. Previous studies showed that 2D CNNs trained on Kinetics achieved better performance [15]. However, it is not trivial to train deep 3D CNNs or better based on the dataset, and we still need to use popular benchmarks that are free of video noise. The results of this study, we found that 3D CNNs are more effective, and can be expected to facilitate further progress in computer vision.

Table 4: Accuracies on the UCF-101 and HMDB-51. All accuracies are averaged over ten splits.

Model	UCF-101	HMDB-51
ResNet-10	42.4	17.1
ResNet-34	50.4	24.4
ResNet-50	57.7	30.4
ResNet-101	58.3	31.0
ResNet-152	58.6	31.4
Two-stream [20]	58.6	31.4
ResNet-101	58.6	31.4
ResNet-152	58.6	31.4
Two-stream [20]	58.6	31.4

in the field experiment, we examined the fine-tuning of Kinetics pre-trained on UCF-101 and HMDB-51. We found that pre-training on large-scale datasets is an effective way to achieve good performance in small datasets, which is one of the most successful architectures in image classification, provides shorter connections that allow a signal to bypass one layer and move to the next layer in the sequence. Since these connections pass through the network, they can facilitate the training of very deep networks.

Next, we applied the various ResNet-based architectures with 2D convolutions used in this study. ResNet, which is one of the most successful architectures in image classification, provides shorter connections that allow a signal to bypass one layer and move to the next layer in the sequence. Since these connections pass through the network, they can facilitate the training of very deep networks.

Next, we applied the various ResNet-based architectures with 2D convolutions used in this study. ResNet, which is one of the most successful architectures in image classification, provides shorter connections that allow a signal to bypass one layer and move to the next layer in the sequence. Since these connections pass through the network, they can facilitate the training of very deep networks.

Next, we applied the various ResNet-based architectures with 2D convolutions used in this study. ResNet, which is one of the most successful architectures in image classification, provides shorter connections that allow a signal to bypass one layer and move to the next layer in the sequence. Since these connections pass through the network, they can facilitate the training of very deep networks.

Next, we applied the various ResNet-based architectures with 2D convolutions used in this study. ResNet, which is one of the most successful architectures in image classification, provides shorter connections that allow a signal to bypass one layer and move to the next layer in the sequence. Since these connections pass through the network, they can facilitate the training of very deep networks.

Next, we applied the various ResNet-based architectures with 2D convolutions used in this study. ResNet, which is one of the most successful architectures in image classification, provides shorter connections that allow a signal to bypass one layer and move to the next layer in the sequence. Since these connections pass through the network, they can facilitate the training of very deep networks.

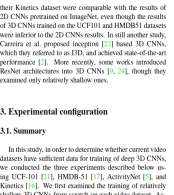


Figure 3: Block of each architecture. We represent conv, β , γ at the kernel size, and the number of feature maps of the convolutional layer as $x \times y \times z$, respectively, and group is the number of groups of group convolutions, which divide the feature maps into small groups. BN refers to batch normalization [11]. Shorter connections of the architectures are abbreviated except for those of DenseNet, which all have no connections.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

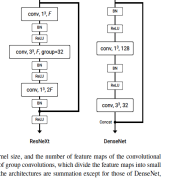


Figure 3: Block of each architecture. We represent conv, β , γ at the kernel size, and the number of feature maps of the convolutional layer as $x \times y \times z$, respectively, and group is the number of groups of group convolutions, which divide the feature maps into small groups. BN refers to batch normalization [11]. Shorter connections of the architectures are abbreviated except for those of DenseNet, which all have no connections.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [11] and a ReLU [1]. Split-synchronous down-sampling is performed by conv1, conv2, and conv5, with a stride of two, except for DenseNet. P is the number of feature channels in convolutional layers, and N is the number of blocks in each layer. DenseNet down-samples splits into the transition layer, that consists of $1 \times 3 \times 3$ convolutional layer and $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv2, conv3, and conv5, and F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $1 \times 3 \times 3$ max-pooling layer with a stride of two is added at the end of each layer. In addition, pre-activation ResNet uses group convolutions with a spatial stride of two. C of the fully-connected layer is the number of classes.

Training. We use multi-scale gradient descent with momentum to train the networks and randomly generate training samples from videos in training data in order to perform data augmentation. First, we select a temporal position in a video. A 16-frame clip is then generated around the selected temporal position. If the video is shorter than 16 frames, then we loop it for many times into 16 frames. Next, we randomly select a spatial position from the corners of the center. In addition to the spatial position, we also select a spatial scale of the sample in order to perform multi-scale augmentation. The procedure used is the same as [3]. The scale is selected from $\{1, 1.25, 1.5, 2\}$. Scale 1 means that the sample width and height are the same as the short side length of the frame, and 2×2 means that the sample is half-size of the short side length. The sample aspect ratio is 1, and the sample is a square-temporally centered in the position, scale, and aspect ratio. We spatially crop the sample at $1/2 \times 1/2$ pixels. The size of each sample is 3 channels, 16 frames $\times 112$ pixels $\times 112$ pixels, and each sample is horizontally flipped with 50% probability. We also perform motion blur, which means that we subtract the mean value of Acrop from the sample for each video channel. All general samples contain the same class labels as their original videos.

In our training, we use cross-entropy losses and back-propagate the gradients. The training parameters include a weight decay of 0.001 and 0.01 for momentum. We perform learning rate decay, we start from a learning rate of 0.1, and divide it by 10 after the validation loss does not improve for 10 epochs. We then start from a learning rate of 0.01, and divide it by 10 after the validation loss does not improve for 10 epochs. We then start from a learning rate of 0.001, and divide it by 10 after the validation loss does not improve for 10 epochs. We then start from a learning rate of 0.0001, and divide it by 10 after the validation loss does not improve for 10 epochs. We then start from a learning rate of 0.00001, and divide it by 10 after the validation loss does not improve for 10 epochs. We then start from a learning rate of 0.000001, and divide it by 10 after the validation loss does not improve for 10 epochs. We then start from a learning rate of 0.0000001, and divide it by 10 after the validation loss does not improve for 10 epochs. We then start from a learning rate of 0.00000001, and divide it by 10 after the validation loss does not improve for 10 epochs. We then start from a learning rate of 0.000000001, and divide it by 10 after the validation loss does not improve for 10 epochs. We then start from a learning rate of 0.0000000001, and divide it by 10 after the validation loss does not improve for 10 epochs. We then start from a learning rate of 0.00000000001, and divide it by 10 after the validation loss does not improve for 10 epochs. We then start from a learning rate of 0.000000000001, and divide it by 10 after the validation loss does not improve for 10 epochs. We then start from a learning rate of 0.0000000000001,

気をつけていること

森を見る，木を見る

- － 森：ざっと全体を通して見る
 - 速読ならこれで十分
 - 背景，イントロ，図表，結果，結論を中心に
- － 木：細かいところまで目を通す（但し，目的を見失わない）
 - どんな情報が欲しいかを明確にして読む
 - 実装したい？ 輪講資料を作りたい？

森を見る (速読)

アブストラクトを見る

ー 良いアブストラクトだとなんとなく全体がわかる

Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet?

Kensho Hara, Hirokatsu Kataoka, Yutaka Satoh
National Institute of Advanced Industrial Science and Technology (AIST)
Tsukuba, Ibaraki, Japan
(kensho.hara, hirokatsu.kataoka, yu.satou@aist.go.jp)

Abstract

The purpose of this study is to determine whether current video datasets have sufficient data for training very deep convolutional neural networks (CNNs) with spatio-temporal three-dimensional (3D) kernels. Recently, the performance levels of 3D CNNs in the field of action recognition have improved significantly. However, to date, conventional research has only explored relatively shallow 3D architectures. We examine the architectures of various 3D CNNs from relatively shallow to very deep ones on current video datasets. Based on the results of those experiments, the following conclusions could be obtained: (i) ResNet-18 training resulted in significant overfitting for UCF-101, HMDB-51, and ActivityNet but not for Kinetics. (ii) The Kinetics dataset has sufficient data for training of deep 3D CNNs, and enables training of up to 152 ResNets layers, interestingly similar to 2D ResNets on ImageNet. ResNeXt-101 achieved 78.4% average accuracy on the Kinetics test set. (iii) Kinetics pre-trained simple 3D architectures outperforms complex 2D architectures, and the pretrained ResNeXt-101 achieved 94.5% and 70.2% on UCF-101 and HMDB-51, respectively. The use of 2D CNNs trained on ImageNet has produced significant progress in various tasks in image. We believe that using deep 3D CNNs together with Kinetics will retrace the successful history of 2D CNNs and ImageNet, and stimulate advances in computer vision for images. The codes and pretrained models used in this study are publicly available¹.

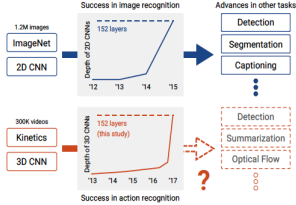


Figure 1: Recent advances in computer vision for images (top) and videos (bottom). The use of very deep 2D CNNs trained on ImageNet generates outstanding progress in image recognition as well as in various other tasks. Can the use of 3D CNNs trained on Kinetics generates similar progress in computer vision for videos?

The Kinetics validation set. Average is for Top-1 and Top-5.	
Model	Top-1
ResNet-101	54.2
ResNet-152	66.1
ResNet-101	60.1
ResNet-152	72.2
ResNet-101	62.8
ResNet-152	73.3
ResNet-101	60.3
ResNet-152	72.7
ResNet-101	63.1
ResNet-152	74.4
ResNet-101	64.1
ResNet-152	74.7
ResNet-101	65.1
ResNet-152	75.4
ResNet-101	65.3
ResNet-152	76.1
ResNet-101	65.3
ResNet-152	76.1
ResNet-101	65.3
ResNet-152	76.1
ResNet-101	65.3
ResNet-152	76.1

Top-1 and Top-5 for Average	
Model	Top-1
ResNet-101	54.2
ResNet-152	66.1
ResNet-101	60.1
ResNet-152	72.2
ResNet-101	62.8
ResNet-152	73.3
ResNet-101	60.3
ResNet-152	72.7
ResNet-101	63.1
ResNet-152	74.4
ResNet-101	64.1
ResNet-152	74.7
ResNet-101	65.1
ResNet-152	75.4
ResNet-101	65.3
ResNet-152	76.1
ResNet-101	65.3
ResNet-152	76.1
ResNet-101	65.3
ResNet-152	76.1
ResNet-101	65.3
ResNet-152	76.1

¹<https://github.com/kenshohara/3D-ResNets-Pytorch>

1. Introduction

The use of large-scale datasets is extremely important when using deep convolutional neural networks (CNNs), which have massive parameter numbers, and the use of CNNs in the field of computer vision has expanded significantly in recent years. ImageNet [4], which includes more

than a million images, has contributed substantially to the creation of successful vision-based algorithms. In addition to such large-scale datasets, a large number of algorithms, such as residual learning [10], have been used to improve image classification performance by adding increased depth to CNNs, and the use of very deep CNNs trained on ImageNet have facilitated the acquisition of generic feature representation. Using such feature representation, in turn, has significantly improved the performance of several other tasks including object detection, semantic segmentation, and image captioning (see top row in Figure 1).

To date, the video datasets available for action recognition have been relatively small when compared with image recognition datasets. Representative video datasets, such as UCF-101 [21] and HMDB-51 [17], can be used to provide realistic videos with sizes around 10 K, but even though they are still used as standard benchmarks, such datasets are obviously too small to be useful for optimizing CNN representations from scratch. In the last couple of years, ActivityNet [5], which

including 20,000 action instances, ActivityNet also provides some additional tasks, such as unsupervised classification and detection, but the number of action instances is still on the order of tens of thousands. This year (2017), it is still difficult to create a successful pretrained model. Kay et al. released the Kinetics dataset [14]. The Kinetics dataset includes more than 300,000 trained videos covering 400 classes. In order to determine whether it can train deep 3D CNNs, we performed a number of experiments using these recent datasets, as well as the UCF-101 and HMDB-51 datasets. Other large datasets, such as Sports-1M [13] and YouTube-8M [1] have been proposed. Although these datasets are larger than Kinetics, their annotations are slightly noisy and only video-level labels have been assigned. In other words, they include frames that do not relate to target actions. Such noise and the presence of unrelated frames have the potential to perturb these models from proceeding with training. In addition, with the size of tens of 10 TB, their scales are simply too large to allow them to be utilized easily. Because of this, we will refrain from discussing these datasets in this study.

2. Action Recognition Approaches

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In this study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [12], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance [6, 7, 22, 23, 24].

Until the aforementioned approaches, we focused on CNNs with 3D convolutional layers, which have been proposed in 3D convolutional CNNs for the use of large-scale video datasets. The results of this study, however, suggest that 3D CNNs can be trained on the same data as 2D CNNs, and that 3D CNNs can be trained on the same data as 2D CNNs. This is because such 3D convolution can be used to extract spatio-temporal features from raw videos. For example, Ji et al. proposed applying 3D convolution to extract spatio-temporal features from raw videos. While Tan et al. trained 3D CNNs, which they referred to as C3D, using the Sports-1M dataset [13]. Since that study, C3D has been seen as a standard baseline for 3D CNNs. In this study, we found that a $3 \times 3 \times 3$ convolutional kernel learned the best performance level for both the Kinetics dataset and the UCF-101 dataset. This result is consistent with the results of previous studies [12, 25]. These authors also found that using optical flow frames in 3D CNNs resulted in higher level of performance than can be achieved from RGB inputs, but that the best performance could be achieved by combining RGB and optical flows. Meanwhile, Kay et al. showed that the results of 3D CNNs trained from scratch on

their Kinetics dataset were comparable with the results of 2D CNNs pretrained on ImageNet, even though the results of 3D CNNs trained on the UCF-101 and HMDB-51 datasets were inferior to those of 2D CNNs. In this study, we trained a 3D CNN on the Kinetics dataset, and we compared the results of this CNN with those of 2D CNNs trained on ImageNet. We first examined the training of relatively shallow 3D CNNs from scratch on Kinetics dataset. After that, we trained a 3D CNN on the Kinetics dataset, and we compared the results of this CNN with those of 2D CNNs trained on ImageNet. We first examined the training of relatively shallow 3D CNNs from scratch on Kinetics dataset. After that, we trained a 3D CNN on the Kinetics dataset, and we compared the results of this CNN with those of 2D CNNs trained on ImageNet.

3. Experimental configuration

3.1. Summary

In this study, to determine whether current video datasets have sufficient data for training of deep 3D CNNs, we conducted the three experiments described below including UCF-101 [21], HMDB-51 [17], ActivityNet [5], and Kinetics [14]. We first examined the training of relatively shallow 3D CNNs from scratch on Kinetics dataset. After that, we trained a 3D CNN on the Kinetics dataset, and we compared the results of this CNN with those of 2D CNNs trained on ImageNet. We first examined the training of relatively shallow 3D CNNs from scratch on Kinetics dataset. After that, we trained a 3D CNN on the Kinetics dataset, and we compared the results of this CNN with those of 2D CNNs trained on ImageNet.

3.2. Network architectures

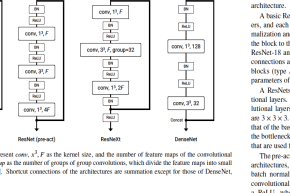
Next, we applied the various ResNet-based architectures with 2D convolutions used in this study. ResNet, which is one of the most successful architectures in image classification, provides shorter connections that allow a signal to bypass one or more layers to the next layer in the sequence. Since these connections pass through the network as gradient flows from the later layers to the early layers, they

Table 4: Top-1 accuracies on UCF-101 and HMDB-51. All accuracies are averaged over ten splits.	
Model	Top-1
ResNet-101	54.2
ResNet-152	66.1
ResNet-101	60.1
ResNet-152	72.2
ResNet-101	62.8
ResNet-152	73.3
ResNet-101	60.3
ResNet-152	72.7
ResNet-101	63.1
ResNet-152	74.4
ResNet-101	64.1
ResNet-152	74.7
ResNet-101	65.1
ResNet-152	75.4
ResNet-101	65.3
ResNet-152	76.1
ResNet-101	65.3
ResNet-152	76.1
ResNet-101	65.3
ResNet-152	76.1

two-streams [12], which utilizes sparse two-stream 3D convolutions pretrained on Kinetics, achieved the best accuracy. However, in our results, we can conclude that using 3D architectures pretrained on Kinetics requires complex 2D architectures. We believe that development of 3D CNNs, which are capable of learning from scratch, is important for vision recognition and its related tasks.

5. Conclusion

In this study, we examined the architectures of various CNNs with spatio-temporal 3D convolutional kernels on current video datasets. Based on the results of those experiments, the following conclusions could be obtained: (i) ResNet-18 training resulted in significant overfitting for UCF-101, HMDB-51, and ActivityNet but not for Kinetics. (ii) The Kinetics dataset has sufficient data for training of deep 3D CNNs, and enables training of up to 152 ResNets layers, interestingly similar to 2D ResNets on ImageNet. ResNeXt-101 achieved 78.4% average accuracy on the Kinetics test set. (iii) Kinetics pre-trained simple 3D architectures outperforms complex 2D architectures, and the pretrained ResNeXt-101 achieved 94.5% and 70.2% on UCF-101 and HMDB-51, respectively. The use of 2D CNNs trained on ImageNet has produced significant progress in various tasks in image. We believe that using deep 3D CNNs together with Kinetics will retrace the successful history of 2D CNNs and ImageNet, and stimulate advances in computer vision for images. The codes and pretrained models used in this study are publicly available¹.



3.1. Summary

In this study, to determine whether current video datasets have sufficient data for training of deep 3D CNNs, we conducted the three experiments described below including UCF-101 [21], HMDB-51 [17], ActivityNet [5], and Kinetics [14]. We first examined the training of relatively shallow 3D CNNs from scratch on Kinetics dataset. After that, we trained a 3D CNN on the Kinetics dataset, and we compared the results of this CNN with those of 2D CNNs trained on ImageNet. We first examined the training of relatively shallow 3D CNNs from scratch on Kinetics dataset. After that, we trained a 3D CNN on the Kinetics dataset, and we compared the results of this CNN with those of 2D CNNs trained on ImageNet.

3.2. Network architectures

Next, we applied the various ResNet-based architectures with 2D convolutions used in this study. ResNet, which is one of the most successful architectures in image classification, provides shorter connections that allow a signal to bypass one or more layers to the next layer in the sequence. Since these connections pass through the network as gradient flows from the later layers to the early layers, they

Table 4: Top-1 accuracies on UCF-101 and HMDB-51. All accuracies are averaged over ten splits.	
Model	Top-1
ResNet-101	54.2
ResNet-152	66.1
ResNet-101	60.1
ResNet-152	72.2
ResNet-101	62.8
ResNet-152	73.3
ResNet-101	60.3
ResNet-152	72.7
ResNet-101	63.1
ResNet-152	74.4
ResNet-101	64.1
ResNet-152	74.7
ResNet-101	65.1
ResNet-152	75.4
ResNet-101	65.3
ResNet-152	76.1
ResNet-101	65.3
ResNet-152	76.1
ResNet-101	65.3
ResNet-152	76.1

two-streams [12], which utilizes sparse two-stream 3D convolutions pretrained on Kinetics, achieved the best accuracy. However, in our results, we can conclude that using 3D architectures pretrained on Kinetics requires complex 2D architectures. We believe that development of 3D CNNs, which are capable of learning from scratch, is important for vision recognition and its related tasks.

5. Conclusion

In this study, we examined the architectures of various CNNs with spatio-temporal 3D convolutional kernels on current video datasets. Based on the results of those experiments, the following conclusions could be obtained: (i) ResNet-18 training resulted in significant overfitting for UCF-101, HMDB-51, and ActivityNet but not for Kinetics. (ii) The Kinetics dataset has sufficient data for training of deep 3D CNNs, and enables training of up to 152 ResNets layers, interestingly similar to 2D ResNets on ImageNet. ResNeXt-101 achieved 78.4% average accuracy on the Kinetics test set. (iii) Kinetics pre-trained simple 3D architectures outperforms complex 2D architectures, and the pretrained ResNeXt-101 achieved 94.5% and 70.2% on UCF-101 and HMDB-51, respectively. The use of 2D CNNs trained on ImageNet has produced significant progress in various tasks in image. We believe that using deep 3D CNNs together with Kinetics will retrace the successful history of 2D CNNs and ImageNet, and stimulate advances in computer vision for images. The codes and pretrained models used in this study are publicly available¹.

3.3. Implementation

Training. We use mini-batch gradient descent with mini-batch size of 32. The training data is randomly generated training samples from videos in training data in order to perform data augmentation. First, we select a temporal position in a video. A 16-frame clip is then generated around the selected temporal position. If the video is shorter than 16 frames, we stop to loop it to make it longer than 16 frames. We randomly select a spatial position from the C -corner of the center. In addition to the spatial position, we also select a spatial scale of the sample in order to perform multi-scale cropping. The procedure used is the same as [23].

Architecture. A basic ResNet block consists of two convolutional layers, and each convolutional layer is followed by batch normalization and ReLU. A shortcut pass connects the top of the block to the layer just before the last ReLU in the block. ResNeXt-101 and ResNeXt-152 use the same identity connections, and zero padding for the shortcut of the basic blocks (type A in [10]) to avoid increasing the number of parameters of these relatively shallow networks. A ResNeXt block consists of three convolutional layers. The kernel sizes of the first and final convolutional layers are 1×1 , whereas those of the second are 3×3 . The shortcut pass of this block is the same as that of the basic block. ResNeXt-101, 152, and 201 adopt the bottleneck. We use identity connections for those blocks that are used for increasing dimensions (type B in [10]).

The pre-activation ResNet is similar to bottleneck ResNet architectures, but there are differences in the connections, batch normalization, and ReLU order. In ResNeXt, each convolutional layer is followed by batch normalization and ReLU. A shortcut pass connects the top of the block to the layer just after the last convolutional layer in the block. In our study, we found that each pre-activation facilitates optimization in the training and reduces overfitting [11]. In this study, pre-activation ResNeXt-201 was evaluated.

WGRN architecture. The same as the ResNet (see the text), but there are differences in the number of ResNet layers for each convolutional layer. WGRN increases the number of ResNet layers in the middle of the network. When training the networks from scratch, we start from learning rate 0.1, and divide it by 10 after the validation loss increases. When performing the tuning, we start from a learning rate of 0.1, and divide it by 10 after the validation loss increases. We use a weight decay of $1e-5$.

ResNeXt architecture. ResNeXt is a different design from ResNet and wider. Unlike the original bottleneck block, the ResNeXt block introduces group convolutions, which divide the feature maps into small groups. These groups are then processed by the convolutional layers in the block. In our study, we found that each pre-activation facilitates optimization in the training and reduces overfitting [11]. In this study, pre-activation ResNeXt-201 was evaluated.

As standard, this study focuses on four datasets: UCF-101 [21], HMDB-51 [17], ActivityNet [5], and Kinetics [14]. We trained the networks with the same learning rate 0.1, and divide it by 10 after the validation loss increases. When performing the tuning, we start from a learning rate of 0.1, and divide it by 10 after the validation loss increases. We use a weight decay of $1e-5$.

ResNeXt architecture. ResNeXt is a different design from ResNet and wider. Unlike the original bottleneck block, the ResNeXt block introduces group convolutions, which divide the feature maps into small groups. These groups are then processed by the convolutional layers in the block. In our study, we found that each pre-activation facilitates optimization in the training and reduces overfitting [11]. In this study, pre-activation ResNeXt-201 was evaluated.

As standard, this study focuses on four datasets: UCF-101 [21], HMDB-51 [17], ActivityNet [5], and Kinetics [14]. We trained the networks with the same learning rate 0.1, and divide it by 10 after the validation loss increases. When performing the tuning, we start from a learning rate of 0.1, and divide it by 10 after the validation loss increases. We use a weight decay of $1e-5$.

森を見る (速読)

イントロ, 概念図を見よう

— どんな問題設定? その研究の新規性とは?

Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet?

Kensho Hara, Hirokatsu Kataoka, Yutaka Satoh
National Institute of Advanced Industrial Science and Technology (AIST)
Tsukuba, Ibaraki, Japan
(kensho.hara, hirokatsu.kataoka, yu.satou@aist.go.jp)

Abstract

The purpose of this study is to determine whether current video datasets have sufficient data for training very deep convolutional neural networks (CNNs) with spatio-temporal three-dimensional (3D) kernels. Recently, the performance levels of 3D CNNs in the field of action recognition have improved significantly. However, to date, conventional research has only explored relatively shallow 3D architectures. We examine the architectures of various 3D CNNs from relatively shallow to very deep ones on current video datasets. Based on the results of those experiments, the following conclusions could be obtained: (i) ResNet-18 training resulted in significant overfitting for UCF-101, HMDB-51, and ActivityNet, but not for Kinetics. (ii) The Kinetics dataset has sufficient data for training of deep 3D CNNs, and enables training of up to 152 ResNets layers, interestingly similar to 2D ResNets on ImageNet. ResNeXt-101 achieved 78.4% average accuracy on the Kinetics test set. (iii) Kinetics pre-trained simple 3D architectures outperforms complex 2D architectures, and the pretrained ResNeXt-101 achieved 94.5% and 70.2% on UCF-101 and HMDB-51, respectively. The use of 2D CNNs trained on ImageNet has produced significant progress in various tasks in image. We believe that using deep 3D CNNs together with Kinetics will retrace the success history of 2D CNNs and ImageNet, and stimulate advances in computer vision for videos. The codes and pretrained models used in this study are publicly available¹.

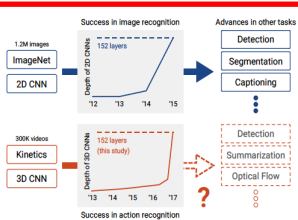


Figure 1: Recent advances in computer vision for images (top) and videos (bottom). The use of very deep 2D CNNs trained on ImageNet generates outstanding progress in image recognition as well as in various other tasks. Can the use of 3D CNNs trained on Kinetics generates similar progress in computer vision for videos?

more than a million images, has contributed substantially to the creation of successful vision-based algorithms. In addition to such large-scale datasets, a large number of algorithms, such as residual learning [10], have been used to improve image classification performance by adding increased depth to CNNs, and the use of very deep CNNs trained on ImageNet have facilitated the acquisition of generic feature representation. Using such feature representation, in turn, has significantly improved the performance of several other tasks including object detection, semantic segmentation, and image captioning (see top row in Figure 1).

To date, the video datasets available for action recognition have been relatively small when compared with image recognition datasets. Representative video datasets, such as UCF-101 [21] and HMDB-51 [17], can be used to provide realistic videos with sizes around 10 K, but even though they are still used as standard benchmarks, such datasets are obviously too small to be useful for optimizing CNN representations from scratch. In the last couple of years, ActivityNet [5], which

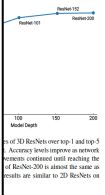


Figure 2: Success in image recognition. The use of very deep 2D CNNs trained on ImageNet generates outstanding progress in image recognition as well as in various other tasks. Can the use of 3D CNNs trained on Kinetics generates similar progress in computer vision for videos?

including 250,000 action instances, ActivityNet also provides some additional tasks, such as unsupervised classification and detection, but the number of action instances is still on the order of tens of thousands. This year (2017), it is of interest to create a sufficiently pretrained model. Kay et al. released the Kinetics dataset [14]. The Kinetics dataset includes more than 300,000 untrimmed videos covering 400 classes. In order to determine whether it can train deep 3D CNNs, we performed a number of experiments using these recent datasets, as well as the UCF-101 and HMDB-51 datasets.

Other large datasets, such as ActivityNet [13] and YouTube8M [1] have been proposed. Although these datasets are larger than Kinetics, their annotations are slightly sparser and only video-level labels have been provided. In other words, they include frames that do not relate to target actions. Such noise and the presence of unrelated frames have the potential to prevent these models from proceeding good training. In addition, with the size of sizes of 10 TB, their scales are simply too large to allow them to be utilized easily. Because of this, we will not refrain from discussing these datasets in this study.

2. Action Recognition Approaches

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

Until the above-mentioned approaches, we focused on CNNs with 3D convolutional layers, which have been proposed in 3D CNNs through the use of large-scale video datasets. In this study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

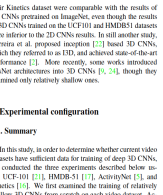


Figure 3: Success in action recognition. The use of very deep 2D CNNs trained on ImageNet generates outstanding progress in image recognition as well as in various other tasks. Can the use of 3D CNNs trained on Kinetics generates similar progress in computer vision for videos?

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

Figure 3: Success in action recognition. The use of very deep 2D CNNs trained on ImageNet generates outstanding progress in image recognition as well as in various other tasks. Can the use of 3D CNNs trained on Kinetics generates similar progress in computer vision for videos?

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

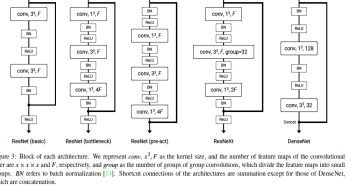


Figure 3: Success in action recognition. The use of very deep 2D CNNs trained on ImageNet generates outstanding progress in image recognition as well as in various other tasks. Can the use of 3D CNNs trained on Kinetics generates similar progress in computer vision for videos?

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

Figure 3: Success in action recognition. The use of very deep 2D CNNs trained on ImageNet generates outstanding progress in image recognition as well as in various other tasks. Can the use of 3D CNNs trained on Kinetics generates similar progress in computer vision for videos?

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

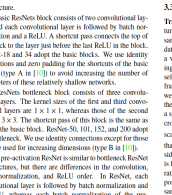


Figure 3: Success in action recognition. The use of very deep 2D CNNs trained on ImageNet generates outstanding progress in image recognition as well as in various other tasks. Can the use of 3D CNNs trained on Kinetics generates similar progress in computer vision for videos?

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

Figure 3: Success in action recognition. The use of very deep 2D CNNs trained on ImageNet generates outstanding progress in image recognition as well as in various other tasks. Can the use of 3D CNNs trained on Kinetics generates similar progress in computer vision for videos?

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

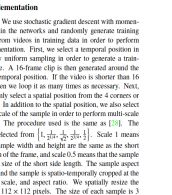


Figure 3: Success in action recognition. The use of very deep 2D CNNs trained on ImageNet generates outstanding progress in image recognition as well as in various other tasks. Can the use of 3D CNNs trained on Kinetics generates similar progress in computer vision for videos?

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

Figure 3: Success in action recognition. The use of very deep 2D CNNs trained on ImageNet generates outstanding progress in image recognition as well as in various other tasks. Can the use of 3D CNNs trained on Kinetics generates similar progress in computer vision for videos?

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

One of the popular approaches to CNN-based action recognition is to train a two-stream CNNs with 2D convolutional layers. In their study, Simonyan et al. proposed a method to use RGB and stacked optical flow frames as appearance and motion information, respectively [18], and showed that combining the two-streams has the ability to improve action recognition accuracy. Since that study, numerous methods based on the two-stream CNNs have been proposed to improve action recognition performance (e.g., [2, 22, 23]).

Temporal 3D CNNs Retrace the History of 2D CNNs and ImageNet?

ポイントは音外と限られない

– In this study (paper/research), ~ や Our contribution(s)~ など序盤に現れる

Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet?

Kensho Hara, Hirokatsu Kataoka, Yutaka Satoh
National Institute of Advanced Industrial Science and Technology (AIST)
Tsukuba, Ibaraki, Japan
{kensho.hara, hirokatsu.kataoka, yu.satoh}@aist.go.jp

Abstract

[illegible]

Figure 1: Recent advances in computer vision for images (top) and videos (bottom). The use of very deep 2D CNNs trained on ImageNet generates outstanding progress in image recognition as well as in various other tasks. Can the use of 3D CNNs trained on Kinetics generates similar progress in computer vision for videos?

For action recognition, CNNs with spatio-temporal three-dimensional (3D) convolutional kernels (3D CNNs) are frequently more effective than CNNs with two-dimensional (2D) kernels [2]. From several years ago [14], 3D CNNs are used for action recognition. However, even the usage of well-organized models [25, 26] has failed to overcome the advantages of 2D-based CNNs that combine both stacked flow and RGB images [30]. The primary reason for this failure has been the lack of an effective method for learning the parameters that are able for optimizing the immense number of parameters in 3D CNNs, which are much larger than those of 2D CNNs. In addition, basically, 3D CNNs can only be trained on video sequences, which are not available for most of the tasks. Recently, however, Carreira and Zisserman achieved a significant breakthrough using the Kinetics dataset as well as the inflation of 2D kernels pretrained on ImageNet into 3D ones [2]. Thus, we now have the benefit of a sophisticated 2D CNN for learning the parameters of 3D CNNs. In this paper, we call 3D CNNs that use the pre-trained 2D CNN as 3D CNNs with 2D CNNs (3D CNNs with 2D CNNs).

Figure 2: Averaged accuracies of 3D ResNets over top-1 and top-5 of the Kinetics validation set. Accuracy increases almost as network depth increases. The improvements obtained until reaching a depth of 152. The accuracy of ResNet-200 is almost the same as that of ResNet-152. These results are similar to 2D ResNets on ImageNet [10].

(of Tube-EM [1]) have been proposed. Although these databases are larger than Kinetics, their annotations are slightly noisier and only video-level labels have been assigned. In other words, they include frames that do not correlate to target actions). Such noise and the presence of unrelated frames have the potential to prevent these models from providing good training. In addition, with the sizes in excess of 10 TB, their scales are simply too large to allow them to be utilized easily. Because of these issues, we will refrain from discussing these datasets in this study.

3.1. Summary

In this study, in order to determine whether current video datasets have sufficient data for training of deep 3D CNNs, we conducted the three experiments described below using UCF-101 [21], HMDB-51 [17], ActivityNet [5], and Kinetics [6]. We first examined the training of relatively shallow 3D CNNs from scratch on each video dataset. According to previous works [9, 16], 3D CNNs trained on UCF-101, HMDB-51, and ActivityNet do not achieve high

1. Introduction

The use of large-scale datasets is extremely important when using deep convolutional neural networks (CNNs), which have massive parameter numbers, and the use of CNNs in the field of computer vision has expanded significantly in recent years. ImageNet [4], which includes more

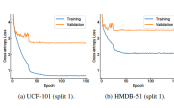


Figure 4: ResNet-18 training and validation losses. The validation high values and were clearly higher than their corresponding train

have been relatively small when compared with image recognition datasets. Representative video datasets, such as UCF-101 [21] and HMDB-51 [17], can be used to provide realistic videos with sizes around 10 K, but even though they are still used as standard benchmarks, such datasets are obviously too small to be useful for optimizing CNN representations from scratch. In the last couple of years, ActivityNet [5], which

In this study, we examine various 3D CNN architectures from relatively shallow to very deep ones using the Kinetics and other popular video datasets (UCF-101, HMDB-51, and ActivityNet) in order to provide us insights for answering the above question. The 3D CNN architectures tested in this study are based on residual networks (ResNets) [10] and their extended versions [11, 12, 30, 31] because they have simple and effective structures. Accordingly, using those datasets, we performed several experiments aimed at

The HMDB-51 [17] and UCF-101 [21] datasets are among the most successful in the field of action recognition. These datasets gained significant popularity in the early years of the field, and are still used as popular benchmarks. However, a recent consensus has emerged that indicates they are simply not large enough for training deep CNNs from scratch [16].

ActivityN
the above
this study

are based

trimmed. The average duration of each video is about 3 seconds. Three train-test splits (70% training and 30% testing) are provided in this dataset.

ActivityNet (v1.3) provides samples from 200 human action classes. ActivityNet consists of 137 untrimmed videos per class. At 141 ActivityNet videos are divided into three classes: action, non-action, and background. Unlike the other datasets, ActivityNet consists of untrimmed videos, which include non-action frames. The total video length is 849 hours, and the total number of action instances is 28,108. The videos are annotated with bounding boxes for training, validation, and testing. More specifically, 50% is used for training, 25% is used for validation, and 25% is used for testing.

The Kinetics dataset has 400 human action classes, and contains more than 600 videos for each class. The videos were temporally trimmed and last around 10 seconds. The total number of the videos is in excess of 300,000. The number of training, validation, and testing sets are about 240,000, 10,000, and 10,000, respectively.

The video properties of all these datasets are similar. Most videos were extracted from YouTube, except for HMDB-51, which includes videos extracted from movies. The videos include dynamic background and camera motion. The number of videos among them are the number of action classes and instances.

Figure 4 shows the training and validation losses of ResNet-18 on each dataset. As can be seen in the figure, the validation losses on UC1-101, HMDB-51, and ActivityNet quickly converged to high values and were clearly higher than the corresponding training losses. These results indicate that the model is overfitting to the training datasets. In addition to those losses, we confirmed per-clip accuracies, which are evaluated for each clip rather than for each video. The validation accuracies of UC1-101, HMDB-51, and ActivityNet are shown in Figure 5. As expected, it should be noted that direct comparisons between our results and those of previous studies would be unfair because the accuracies reported in most papers were per-video accuracies. However, since these accuracies are very low even in our case, we can still compare our results with theirs. It is difficult to train deep 3D CNNs from scratch on UC1-101, HMDB-51, and ActivityNet.

In contrast, the training and validation losses on Kinetics were very similar to each other, and the training losses were slightly higher than the validation losses. These results would lead us to conclude that training ResNet-18 on Kinetics did not result in overfitting, and that it is possible for Kinetics to train deep 3D CNNs. In the next section, we will further investigate deeper 3D CNNs on Kinetics.

the accuracy of ReNet201 was almost the same as the accuracy of ReNet200. The accuracy of ReNet201 was higher than ReNet200 on ImageNet. Interestingly, the results are similar to those for 2D ReNet201 on ImageNet [13]. More specifically, the accuracy of ReNet201 is higher than ReNet200 on the depth increased until reaching the depth of 200. This is because the accuracy of ReNet201 is not sensitive to the depth of the network. These results indicate that the proposed architecture is sufficient data for 3D CNNs.

Comparisons with other architectures are shown in Table 3. The accuracy of the proposed architecture is higher than the accuracy of ReNet201 and slightly low when compared with the standard ReNet200 though He et al. reported that this is due to the difference in the data augmentation. The accuracy of WNet50 is larger than the ReNet-152. Furthermore, the accuracy of the proposed architecture is higher than the other architectures listed. This result is also a result of the proposed architecture. The proposed architecture is a cardinality works well for the 3D ReNet201 on Kinetics. In contrast, the accuracies of the ReNet201 and 200 are lower than the proposed architecture. This is because the proposed architecture does not require data augmentation to train efficiently. The proposed architecture does not need such techniques to transfer efficient 2D CNNs.

Table 3 shows the results of the Kinetics test set using the proposed architecture. The proposed architecture is compared with the state-of-the-art methods. Here, we use the proposed architecture with the proposed data augmentation. We compared with CS3D with batch normalization [10], which is a 3D-layer network, as well as CNN-3STM and two-way fusion [11]. The proposed architecture is compared with 2D-3D networks trained on Kinetics. In contrast, the proposed architecture is compared with the proposed architecture. The proposed architecture performs ReNet201 (even though the ReNet201 is not a depth

Model	Top-1	Top-5	Average
ResNeXt-101	64.2	83.1	68.6
ResNeXt-101 (64f)	60.1	81.9	71.0
ResNeXt-101	61.3	83.1	72.2
ResNeXt-101	62.8	83.9	73.3
ResNeXt-101	61.0	84.4	72.7
ResNeXt-200	63.1	84.4	73.7
ResNeXt-200 (pre-act)	63.8	83.7	73.7
Wide ResNeXt-50	64.1	85.3	74.6
Wide ResNeXt-50	61.0	85.7	73.3
DenseNet-121	61.8	81.9	71.9
DenseNet-201	61.1	83.3	72.3

Model	Top-1	Top-5	Average
ResNeXt-101	57.0	79.9	68.4
ResNeXt-101 (64f)	—	—	74.4
CNN5-LSTM [10]	57.0	79.9	68.4
CNN5+Strm [10]	61.0	81.3	71.2
C3D w/ BN [10]	64.1	79.5	67.8
ResNeXt-101	68.4	80.0	74.2
ResNeXt-101	71.6	90.0	80.8

those data training a

5. Conclusion

In this study, we examined the architectures of various CNNs with spatio-temporal 3D convolutional layers across different video datasets. Based on the results of these experiments, the following conclusions could be obtained: (i) ReSeNet-18 training resulted in the significant overfitting UCF-101, HMDB-51, and ActivityNet but not for Kinetics dataset. (ii) The Kinetics dataset has sufficient data for training deep 3D CNNs, and enables training up to 152 ResNet layers, interestingly similar to 2D ResNets on ImageNet. (iii) Kinetics pre-trained model on 24 activities outperforms the complete 2D architectures on UCF-101 and HMDB-51, the pre-trained ResNet-101 achieved 94.9% and 70.2%.

Figure 1 illustrates the architecture of the proposed model, which consists of five sequential processing blocks:

- ResNet18 (Stage1):** Contains a $\text{conv}_1, 7, F$ layer followed by a max pooling layer.
- ResNet18 (Stage2):** Contains a $\text{conv}_2, 11, F$ layer, a max pooling layer, a $\text{conv}_3, 5, F$ layer, a max pooling layer, a $\text{conv}_4, 3, F$ layer, a max pooling layer, a $\text{conv}_5, 11, 4F$ layer, and a max pooling layer.
- ResNet18 (Stage3):** Contains a $\text{conv}_6, 11, F$ layer, a max pooling layer, a $\text{conv}_7, 11, F$ layer, a max pooling layer, a $\text{conv}_8, 11, F$ layer, a max pooling layer, a $\text{conv}_9, 11, F$ layer, a max pooling layer, a $\text{conv}_{10}, 11, 4F$ layer, and a max pooling layer.
- ResNet18:** Contains a $\text{conv}_{11}, 7, F$ layer, a max pooling layer, a $\text{conv}_{12}, 5, F$ layer, a max pooling layer, a $\text{conv}_{13}, 3, \text{group}=12$ layer, a max pooling layer, a $\text{conv}_{14}, 11, 2F$ layer, a max pooling layer, a $\text{conv}_{15}, 11, 2F$ layer, and a max pooling layer.
- SimpleNet:** Contains a $\text{conv}_{16}, 11, 128$ layer, a max pooling layer, a $\text{conv}_{17}, 11, 128$ layer, a max pooling layer, a $\text{conv}_{18}, 7, 22$ layer, and a max pooling layer.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [13] and a ReLU [18]. Spatio-temporal down-sampling is performed by conv3_1, conv4_1, and conv5_1 with a stride of two, except for DenseNet. F is the number of feature channels corresponding to *Plane*, and M is the number of *Modules* in each hour. TimeNet down-samples feature in the transition hour.

ResNeSts block consists of two convolutional layers and each convolutional layer is followed by batch normalization and a ReLU. A shortcut pass connects the top of the block to the layer just before the last ReLU in the block. ResNet-18 and 34 adopt the basic blocks. We use identity connections and skip connections for the shortcuts of the basic blocks (Type A in [10]) to avoid increasing the number of parameters of these relatively shallow networks.

ResNeSts bottleneck block consists of three convolutional layers. The lateral sizes of the first and third convolutional layers are $1 \times 1 \times 1$, whereas those of the second convolutional layer are $3 \times 3 \times 3$. We use identity connections for the shortcuts of the bottleneck blocks (Type B in [10]).

The pre-activation ResNet is similar to bottleneck ResNet. However, there are differences in the convolutional layer order. The first convolutional layer is followed by batch normalization and a ReLU, whereas each batch normalization of the pre-activation ResNet is followed by the ReLU and a convolutional layer. A shortcut pass connects the top of the block to

3.3. Implementation

We use stochastic gradient descent with momentum to train the networks and generate *positive training samples* from videos in training data in order to perform generalization. First, we select a temporal position in the video to uniformly sample frames to generate samples. For example, a 16-frame clip is then generated around the temporal position. If the video is shorter than 16 frames, then we loop it as many times as necessary. Next, we randomly select a spatial position in the frame to generate a frame. In addition to the spatial position, we also select a scale of the sample in order to perform multi-scale learning. The procedure used is the same as [28]. We select a sample from $\{1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{5}, \frac{1}{6}, \frac{1}{8}, \frac{1}{10}\}$. Scale 1 means the sample width and height are the same as the short side of the frame. Scale 2 means the sample width and height are half of the short side of the frame. The sample aspect ratio is 1 and the sample is spatio-temporally cropped at the top, left, and, aspect ratio. We spatially resize the sample to 112×112 pixels. The size of each sample is 3 or 5 frames, i.e. 16 frames for 112×112 pixels and 8 frames for 56×56 pixels, with 50% probability. We also subsample, with 50% means that we subtract the *ActiveNet* from the sample for each class label. Negative samples retain the same class labels.

ing, we use cross-entropy losses and backgradients. The training parameters include 0.001 and 0.9 for momentum. When works from scratch, we start from learning rate by 10 after the validation loss saturates. In fine tuning, we start from a learning rate of $1e-5$ and a weight decay of $1e-5$.

We adopt the sliding window manner to generate, (i.e., each video is split into non-overlapped), and recognize actions in videos using the clips. Each clip is spatially cropped around a point at scale 1. We then input each clip into the network to estimate the clip class scores, which are average clips of the video. The class that has the highest score indicates the recognized class label.

we, this study focuses on four datasets: UCF-B-51 [17], ActivityNet [5], and Kinetics [16], includes 13,320 action instances from 101 classes. The videos were temporally trimmed action frames. The average duration of each is 7 seconds. Three train/test splits (70% training) are provided in the dataset. It includes 6,766 videos from 51 human action

In this study, we examine various 3D CNN architectures from relatively shallow to very deep ones using the Kinetics and other popular video datasets (UCF-101, HMDB-51, and ActivityNet) in order to provide us insights for answering the above question. The 3D CNN architectures tested in this study are based on residual networks (ResNets) [10] and their extended versions [11, 12, 30, 31] because they have simple and effective structures. Accordingly, using those datasets, we performed several experiments aimed at training and testing those architectures from scratch, as well

4. Results and discussion

4.1. Analyses of training on each dataset

We began by training ResNet-18 on each dataset. According to previous works [9, 16], 3D CNNs trained on UCF-101, HMDB-51, and ActivityNet do not achieve high accuracy whereas ones trained on Kinetics work well. We tried to reproduce such results in this experiment. In this process, we used split 1 of UCF-101 and HMDB-51, and

Here, we will show ResNets accuracies changes based on model depths. Figure 2 shows the averaged accuracies over top-1 and top-5 ones. We can see that, essentially, as the depth increased, accuracies improved, and that the improvements continued until reaching the depth of 152. We can also see that deeper ResNet-152 achieved significant improvement of accuracies compared with ResNet-18, which

of 13D is $3 \times 6 \times 6 \times 224 \times 224$, whereas that of ResNeXt-101 is $3 \times 3 \times 16 \times 112 \times 112$. Thus, 13D is 64 times times larger than ResNeXt-101. To confirm the accuracies when using larger inputs, we also evaluated the ResNeXt-101 that uses $3 \times 64 \times 112 \times 112$ inputs, which are the largest available sizes in our environment (NVIDIA TITAN X 8G). We can see that the network, referred as ResNeXt-101 (64x) in Table 3, outperformed RGB-13D even though the input size is still four times smaller than that of 13D. We can conclude that deeper 3D architectures trained on Kinetics are effective. In addition, it is felt that combining two-stream archi-

itics can train deep 3D CNNs from scratch, but that it is difficult to train such networks on other datasets. In this section, we fine-tune the Kinetics pretrained 3D CNNs UCF-101 and HMDB-51. The results of this experiment are important for determining whether the 3D CNNs are effective for other datasets. It should be noted that, in this experiment, fine-tuning was only performed to train conv5_3 and the fully connected layer because it achieved the best performance during the preliminary experiments.

Table 4 shows the accuracies of Kinetics pretrained 3D CNNs, as well as ResNet-18 trained from scratch, in UCF-

We show the results of our comparison with state-of-the-art methods in Table 5. Here, we can see that ResNeXt-101 achieved higher accuracies compared with C3D [23], 3D [19], two-stream CNN [20], and TDD [27]. Furthermore, we can also see that ResNeXt-101 (64f), which utilize larger inputs described in previous section, slightly outperformed ST Multipler Net [7] and TSN [29], which utilize `torch.nn.Sequential` architecture. We can also see

the advances in video recognition and its related tasks. Following the significant advances in image recognition made by 2D CNNs and ImageNet, pretrained 2D CNNs on ImageNet experienced significant progress in various tasks such as object detection, semantic segmentation, and image captioning. It is felt that, similar to these, 3D CNNs and Kinetics have the potential to contribute to significant progress in fields related to various video tasks such as action detection, video summarization, and optical flow estimation. In the future work, we will investigate transfer learning not only for action recognition but also for other such tasks.

[illegible]

森を見る (速読)

ポイントが意外に限られる

一関車研究の最後に 従来との比較 差分が書かれる (ごとか多い)

Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet?

Kensho Hara, Hirokazu Kataoka, Yutaka Sato
National Institute of Advanced Industrial Science and Technology (AIST)
Tsukuba, Ibaraki, Japan
(hara.kensho, kataoka.hirokazu, sato.yutaka@aist.go.jp)

Abstract

The purpose of this study is to determine whether current video datasets have sufficient data for training very deep convolutional neural networks (CNNs) with spatiotemporal three-dimensional (3D) features. Recently, the performance tests of 3D CNNs in the field of action recognition have improved significantly. However, in this, conventional research has only explained intuitively shallow 2D architectures. We examine the architecture of various 3D CNNs from relatively shallow to very deep ones on current video datasets. Based on the results of these experiments, the following conclusions could be obtained: (1) ReNet-18 training resulted in significant overfitting on UCF-101, HMDB-51, and ActivityNet but not for Kinetics. (2) The Kinetics dataset has sufficient data for training of deep 3D CNNs. (3) The ReNet-18 architecture, unlike the Kinetics test set, can be trained on ImageNet, and the performance of ReNet-18 on ImageNet is 74.9% and 72.9% on UCF-101 and HMDB-51, respectively. The use of 3D CNNs trained on ImageNet has produced significant progress in action recognition. We believe that using deep 3D CNNs together with Kinetics will retrace the prehistory of 2D CNNs and ImageNet, and stimulate advances in action recognition for videos. The code and pretrained models used in this study are publicly available.

1. Introduction

The use of large-scale datasets is extremely important when using deep convolutional neural networks (CNNs), which have intuitive parameter numbers, and the use of CNNs in the field of computer vision has expanded significantly in recent years. ImageNet [1], which includes more

https://github.com/kenshohara/3d_kinetics_finetune

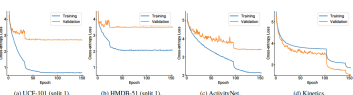


Figure 4: ReNet-18 training and validation losses. The validation losses on UCF-101, HMDB-51, and ActivityNet quickly converged to high values and were clearly higher than their corresponding training losses. The validation losses on Kinetics were slightly higher than the corresponding training losses, significantly different from the other datasets.

trained. The average duration of each video is about 3 seconds. These training splits (CNN training and 30% testing) are provided in this study. The ReNet-18 provides samples from 200 human action classes with an average of 17 annotated videos per class and 11 action instances per video. Unlike the other datasets, this includes videos of untrained videos, which include non-action frames. The total video length is 849 hours, and the total number of action instances is 28,108. This dataset is randomly split into three different subsets: training, validation, and testing. More specifically, 30% is used for training, 25% for validation, and 25% is used for testing.

The Kinetics dataset has 600 human action classes, and consists of more than 400 videos for each class. The videos were temporally trained and last around 10 seconds. The total number of the videos is 10,000. The training, validation, and testing sets are 6,000, 2,000, and 2,000, respectively. The video properties of all three datasets are summarized in Table 1. The videos were extracted from YouTube, except for HMDB-51, which includes videos extracted from Kinetics. The videos include diverse background and camera motion, and the motion difference among them is the number of action classes and instances.

We trained the videos in lengths of 200 pixels without changing their aspect ratios and then used them.

2. Results and discussion

2.1. Analyses of training on each dataset

We began by training ReNet-18 on each dataset. According to previous works [1, 3], 3D CNNs trained on UCF-101, HMDB-51, and ActivityNet do not achieve high accuracy when trained on Kinetics work set. We tried to reproduce such results in this experiment. In this process, we used split 1 of UCF-101 and HMDB-51, which the training and validation sets of ActivityNet and Kinetics.

is a somewhat larger video dataset, has become available, and in this we make it possible to accomplish additional tasks such as supervised action classification and detection, but the number of action instances is constant as still limited. More recently, the Kinetics dataset [14] was created with the aim of being positioned as a large video dataset intended that it is roughly equivalent to the position held by ImageNet in relation to image datasets. More than 300 videos have been collected for the Kinetics dataset, which means that the scale of video datasets has begun to approach that of image datasets.

For action recognition, CNNs with spatiotemporal three-dimensional (3D) convolutional kernels (3D CNNs) are recently more effective than CNNs with two-dimensional (2D) kernels [1]. From several years ago [14, 15], 3D CNNs are thought to be possible as effective tool for action recognition. However, even the usage of well-organized networks [12–15] has failed to overcome the advantage of 2D-based CNNs that combine both spatial and RGB images [12]. The primary reason for this failure has been the relatively small data-scale of video datasets that are available for optimizing the immense number of parameters in 3D CNNs, which are much larger than those of 2D CNNs. In addition, baseline 3D CNNs can only be trained on video datasets, which are much smaller than those of ImageNet. Recently, however, Carreira and Zisserman achieved a significant breakthrough using the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

2.2. Related Work

2.1. Video Datasets

In this study, we examined the ability for action recognition from relatively shallow to very deep 3D CNNs. In order to provide the current video datasets, we compared with image datasets, which include non-action frames, and the use of CNNs in the field of computer vision has expanded significantly in recent years. ImageNet [1], which includes more

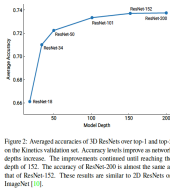


Figure 2: Average accuracy of 3D ReNet-18 on top-1 and top-5 for UCF-101, HMDB-51, and Kinetics. The accuracy of ReNet-18 on ImageNet is 74.9% and 72.9% on UCF-101 and HMDB-51, respectively. The use of 3D CNNs trained on ImageNet has produced significant progress in action recognition. We believe that using deep 3D CNNs together with Kinetics will retrace the prehistory of 2D CNNs and ImageNet, and stimulate advances in action recognition for videos. The code and pretrained models used in this study are publicly available.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

2.2. Related Work

2.1. Video Datasets

In this study, we examined the ability for action recognition from relatively shallow to very deep 3D CNNs. In order to provide the current video datasets, we compared with image datasets, which include non-action frames, and the use of CNNs in the field of computer vision has expanded significantly in recent years. ImageNet [1], which includes more

trained. The average duration of each video is about 3 seconds. These training splits (CNN training and 30% testing) are provided in this study. The ReNet-18 provides samples from 200 human action classes with an average of 17 annotated videos per class and 11 action instances per video. Unlike the other datasets, this includes videos of untrained videos, which include non-action frames. The total video length is 849 hours, and the total number of action instances is 28,108. This dataset is randomly split into three different subsets: training, validation, and testing. More specifically, 30% is used for training, 25% for validation, and 25% is used for testing.

The Kinetics dataset has 600 human action classes, and consists of more than 400 videos for each class. The videos were temporally trained and last around 10 seconds. The total number of the videos is 10,000. The training, validation, and testing sets are 6,000, 2,000, and 2,000, respectively. The video properties of all three datasets are summarized in Table 1. The videos were extracted from YouTube, except for HMDB-51, which includes videos extracted from Kinetics. The videos include diverse background and camera motion, and the motion difference among them is the number of action classes and instances.

We trained the videos in lengths of 200 pixels without changing their aspect ratios and then used them.

2.2. Results and discussion

2.1. Analyses of training on each dataset

We began by training ReNet-18 on each dataset. According to previous works [1, 3], 3D CNNs trained on UCF-101, HMDB-51, and ActivityNet do not achieve high accuracy when trained on Kinetics work set. We tried to reproduce such results in this experiment. In this process, we used split 1 of UCF-101 and HMDB-51, which the training and validation sets of ActivityNet and Kinetics.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

2.2. Related Work

2.1. Video Datasets

In this study, we examined the ability for action recognition from relatively shallow to very deep 3D CNNs. In order to provide the current video datasets, we compared with image datasets, which include non-action frames, and the use of CNNs in the field of computer vision has expanded significantly in recent years. ImageNet [1], which includes more

including 20,000 action instances. ActivityNet also provides some additional tasks, such as unsupervised classification and detection, but the number of action instances is still as the order of tens of thousands. This year (2017), an effort to create a successful pretrained model, Kay et al. released the Kinetics dataset [14]. The Kinetics dataset includes more than 300,000 training videos covering 400 classes, in order to determine whether it can train deep 3D CNNs.

Other large datasets such as Sports-1M [13] and YouTube-8M [1] have been proposed. Although these datasets are larger than Kinetics, their annotations are slightly noisy and only video-level labels have been assigned. In other words, they include frames that do not belong to target actions. Such noise and the presence of unrelated frames have the potential to prevent these models from proceeding good training. In addition, with the size of over 10 TB, these scales are simply too large to store and use as a default setup. Because of these issues, we

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

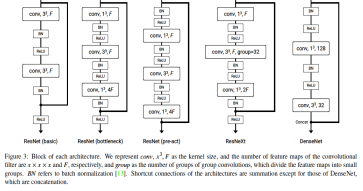


Figure 3: Block of each architecture. We represent conv_i as conv_i at the kernel size, and the number of feature maps of the convolutional filter are $3 \times 3 \times 3$, respectively, and group is the number of groups of group-convolution, which divide the feature maps into small groups. BN refers to batch normalization [1]. Shallow connections of the architectures are annotated except for those of DenseNet, which are not annotated.

Table 1: Network Architectures. Each convolutional layer is followed by batch normalization [1] and a ReLU [10]. Spatio-temporal down-sampling is performed by conv_1 , conv_2 , and conv_3 with a stride of two, except for DenseNet. P is the number of feature channels, which is followed by a $3 \times 3 \times 3$ convolutional layer and a $2 \times 2 \times 2$ average pooling layer with a stride of two, after conv_4 , conv_5 , and conv_6 . F is the number of the number of feature channels of each block in each layer, and N is the same as that of the other networks. A $3 \times 3 \times 3$ max-pooling layer (size 20) is also located before conv_7 of all networks for down-sampling. In addition, each sample is equally down-sampled with a spatial stride of 2. C of the fully-connected layer is the number of classes.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

However, can 3D CNNs retrace the successful history of action recognition? More specifically, can the use of 3D CNNs trained on Kinetics produce significant progress in action recognition and other various tasks? (See bottom right of Figure 2.) In order to answer this question, we trained 3D CNNs on Kinetics to see if they can be used as ImageNet for 2D CNNs. The results of this study, we expect to facilitate further progress in computer vision for videos.

As for the fine-tuning, the results of these experiments (Section 4 for details) show the Kinetics dataset as well as the ImageNet of 2D kernels pretrained on ImageNet to train 3D CNNs [16]. This, we may have the benefit of a substantial 3D convolution that can be engaged by the Kinetics dataset.

木を見る（精読）

速読 + 目的に合わせた読み方を

- 基本的には速読と，部分を詳細に読み込んでいく
- ここから先は何をしたいかに依存する

Can Spontaneous 3D CNNs Reveal the History of 2D CNNs and ImageNet?

Benjamin Hoover, Hidenori Kato, Yuta Saito
National Institute of Advanced Industrial Science and Technology (AIST)
Tsukuba, Japan, hidenori.kato@aist.go.jp

Abstract

The properties of deep convolutional neural networks (CNNs) have been extensively studied in the context of image classification. However, the historical development of CNNs and their relationship to the ImageNet dataset remain largely unknown. In this paper, we propose a novel method to reveal the history of CNNs by analyzing the spontaneous activation patterns of neurons in a 3D CNN. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

1 Introduction

The goal of this work is to reveal the historical development of CNNs and their relationship to the ImageNet dataset. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

2 Related Work

The goal of this work is to reveal the historical development of CNNs and their relationship to the ImageNet dataset. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

2.1 ImageNet

The ImageNet dataset is a large-scale dataset of images for image classification. It contains over 1.2 million images from 1000 classes. The dataset is widely used for training and evaluating image classification models. In this paper, we use the ImageNet dataset to train and evaluate our 3D CNN model. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

2.2 3D CNN

A 3D CNN is a deep convolutional neural network that takes 3D volumes as input. It is used for tasks such as video classification and object detection. In this paper, we use a 3D CNN to analyze the spontaneous activation patterns of neurons in a 3D CNN. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

3 Experimental Setup

The goal of this work is to reveal the historical development of CNNs and their relationship to the ImageNet dataset. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

3.1 Dataset

We use the ImageNet dataset for training and evaluation. The dataset is split into training and validation sets. We use the training set to train our 3D CNN model and the validation set to evaluate its performance. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

3.2 Implementation

We implement our 3D CNN model using TensorFlow. We use a standard 3D CNN architecture with three convolutional layers and a fully connected layer. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

4 Results and Discussion

The goal of this work is to reveal the historical development of CNNs and their relationship to the ImageNet dataset. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

4.1 Spontaneous Activation Patterns

We analyze the spontaneous activation patterns of neurons in a 3D CNN. We show that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN. This suggests that the 3D CNN model captures the historical information of the 2D CNN model. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

4.2 ImageNet Reconstruction

We use the spontaneous activation patterns of neurons in a 3D CNN to reconstruct the ImageNet dataset. We show that the reconstructed ImageNet dataset is highly similar to the original ImageNet dataset. This suggests that the 3D CNN model captures the historical information of the ImageNet dataset. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

5 Conclusion

The goal of this work is to reveal the historical development of CNNs and their relationship to the ImageNet dataset. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

5.1 Summary

We have shown that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN. This suggests that the 3D CNN model captures the historical information of the 2D CNN model and the ImageNet dataset. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

5.2 Future Work

There are several directions for future work. First, we can extend our method to other datasets and tasks. Second, we can investigate the relationship between the spontaneous activation patterns of neurons in a 3D CNN and the activation patterns of neurons in a 2D CNN. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

6 Acknowledgments

We thank the anonymous reviewers for their helpful comments. This work was supported by the Japanese Ministry of Education, Culture, Sports, Science and Technology (MEXT) Grant Number 15K12001.

6.1 Funding

This work was supported by the Japanese Ministry of Education, Culture, Sports, Science and Technology (MEXT) Grant Number 15K12001. We also thank the Japanese Ministry of Education, Culture, Sports, Science and Technology (MEXT) for their support. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

6.2 Ethics Statement

This work does not involve any human subjects or sensitive data. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

7 References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

7.1 ImageNet

[2] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 2015.

7.2 3D CNN

[3] Yinpeng Chen, Bihan Wen, Hao Tang, and Li Fei-Fei. 3D CNN for video classification. *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

8 Appendix

The goal of this work is to reveal the historical development of CNNs and their relationship to the ImageNet dataset. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

8.1 Spontaneous Activation Patterns

We analyze the spontaneous activation patterns of neurons in a 3D CNN. We show that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN. This suggests that the 3D CNN model captures the historical information of the 2D CNN model. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

8.2 ImageNet Reconstruction

We use the spontaneous activation patterns of neurons in a 3D CNN to reconstruct the ImageNet dataset. We show that the reconstructed ImageNet dataset is highly similar to the original ImageNet dataset. This suggests that the 3D CNN model captures the historical information of the ImageNet dataset. We show that the spontaneous activation patterns of neurons in a 3D CNN can be used to reconstruct the history of 2D CNNs and the ImageNet dataset. Our method is based on the observation that the spontaneous activation patterns of neurons in a 3D CNN are highly correlated with the activation patterns of neurons in a 2D CNN.

9 Bibliography

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

9.1 ImageNet

[2] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 2015.

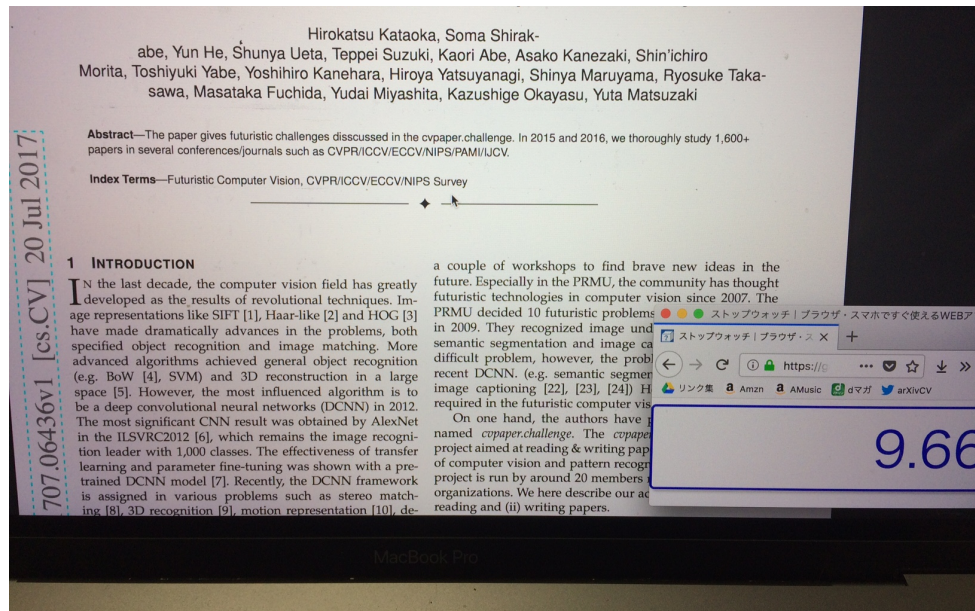
9.2 3D CNN

[3] Yinpeng Chen, Bihan Wen, Hao Tang, and Li Fei-Fei. 3D CNN for video classification. *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

気をつけていること（個人サーベイ）

タイムトライアル

- 時間を気にして，締め切りある読みにする
 - 実際ストップウォッチにおいて論文読んでます！
- 目安時間
 - 速読（15～30分）
 - 精読（1時間～理解できるまで）



気をつけていること（個人サーベイ）

論文サマリを作成しよう

- －（自分だけでなく他の人が）素早く研究の肝をつかむ
- －まとめることで記憶の定着を早くする

【1】  Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, Antonio Torralba, "SUN Database: Large-scale Scene Recognition from Abbey to Zoo", in CVPR2010.

Keywords: Dataset, Scene Categorization, Benchmark, Recognition

概要	データセットの概要
コンピュータビジョンにおいてシーン認識のデータベースである Scene UNderstanding (SUN) databaseを提案。シーン認識の裾野を広げた。	シーン認識に関する397クラス、130,519枚の画像が含まれる。画像例は次ページ。比較した特徴量は、HOG, denseSIFT, self-similarity (ssim), LBP, GIST, textonなど。
新規性・差分	結果
それまでの物体認識のデータセットでは数百クラスの識別クラスが用意されていたが、シーン認識では15種類程度しか含まれていなかった。SUN databaseでは、それまでのデータセットをさらに拡大させ、397クラスのシーンを含む、大規模なデータセットである。	次ページの図の通り。全ての特徴量を統合するのが最も精度が高いことが判明した(38.0%)。次いでHOG2x2 (27.2%), geometry texton hist (23.5%), ssim (22.5%), dense SIFT (21.5%)であった。
Links	
論文ページ: http://cs.brown.edu/~hays/papers/sun.pdf	HOG https://hal.archives-ouvertes.fr/inria-00548512/document GIST http://cvcl.mit.edu/scene_understanding.html SSIM http://www.researchgate.net/profile/Eli_Shechtman/publication/221362526_Matching_Local_Self-Similarities_across_Images_and_Videos/links/02e7e520897af25746000000.pdf DenseSIFT http://www.vision.caltech.edu/Image_Datasets/Caltech101/cvpr06b_lana.pdf LBP http://www.outex.oulu.fi/publications/pami_02_opm.pdf Sparse SIFT http://www.robots.ox.ac.uk/~vgg/publications/papers/sivic04b.pdf Texton http://www.ics.uci.edu/~fowlkes/papers/mftm-iccv01.pdf
プロジェクトページ: http://vision.princeton.edu/projects/2010/SUN/	

まとめの例：概要，新規性，手法，結果，リンク等があるのが望ましい

グループサーベイ

グループサーベイで意識すること

集団の力をうまく利用する

「モチベーション維持」

「集合知」

「作業分割と統合」

等を発揮するためグループで活動するメリットは多い

最近の重要論文は本数が多いので協力して知識を獲得する

(メジャー会議_{のみ}で1,400+本/年, arXivは5,000+本/年?)

論文リストを作成

タスク/進捗を可視化する

- {全体数, 残りの本数} がどれくらいか見えるように
- ノルマはゆるーく決めるが, チェックは確実に
 - タイトに個人を決めると精神的に辛くなるのでカバーし合う

1505	Deep Neural Networks Are Easily Fooled: High Confidence Predictions for Unrecognizable Images Anh Nguyen, Jason Yosinski, Jeff Clune	1505	Small-Variance Nonparametric Clustering on the Hypersphere Julian Straub, Trevor Campbell, Jonathan P. How, John W. Fisher III
----------------------	---	----------------------	---

Monday June 8, 10:10am-12:30pm

Poster Session	
1505	Going Deeper With Convolutions Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich
1506	Propagated Image Filtering Jen-Hao Rick Chang, Yu-Chiang Frank Wang
1506	Web Scale Photo Hash Clustering on A Single Machine Yunchao Gong, Marcin Pawlowski, Fei Yang, Louis Brandy, Lubomir Bourdev, Rob Fergus
1506	Expanding Object Detector's Horizon: Incremental Learning Framework for Object Detection in Videos Alina Kuznetsova, Sung Ju Hwang, Bodo Rosenhahn, Leonid Sigal
1506	Supervised Discrete Hashing Fumin Shen, Chunhua Shen, Wei Liu, Heng Tao Shen
1505	What do 15,000 Object Categories Tell Us About Classifying and Localizing Actions? Mihir Jain, Jan C. van Gemert, Cees G. M. Snoek
1508	Landmarks-Based Kernelized Subspace Alignment for Unsupervised Domain Adaptation Rahaf Aljundi, Rémi Emonet, Damien Muselet, Marc Sebban

CVPR 2015 完全読破チャレンジより

分担して資料を作成してお互いに参照

論文サマリを上手に活用

- 論文を確実にまとめる，を徹底すると後に読む人が楽
 - リストにチェックされている論文はサマリがある
- 素早くポイントをつかんで原論文を読み始める

[1]  Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, Antonio Torralba, "SUN Database: Large-scale Scene Recognition from Abbey to Zoo", in CVPR2010.


Keywords: Dataset, Scene Categorization, Benchmark, Recognition

概要	データセットの概要
コンピュータビジョンにおいてシーン認識のデータベースである Scene Understanding (SUN) databaseを提案。シーン認識の裾野を広げた。	シーン認識に関する397クラス、130,519枚の画像が含まれる。画像例は次ページ。比較した特徴量は、HOG, denseSIFT, self-similarity (ssim), LBP, GIST, textronなど。
新規性・差分	結果
それまでの物体認識のデータセットでは数百クラスの識別クラスが用意されていたが、シーン認識では15種類程度しか含まれていなかった。SUN databaseでは、それまでのデータセットをさらに拡大させ、397クラスのシーンを含む、大規模なデータセットである。	次ページの図の通り。全ての特徴量を統合するのが最も精度が高いことが判明した(38.0%)。次いでHOG2x2 (27.2%), geometry textron hist (23.5%), ssim (22.5%), dense SIFT (21.5%)であった。
Links	HOG https://hal.archives-ouvertes.fr/jrjia-00548512/document GIST http://cvcl.mit.edu/scene_understanding.html SSIM http://www.researchgate.net/profile/Eli_Shechtman/publication/221362526_Matching_Local_Self-Similarities_across_Images_and_Videos/links/02e7e520897ef25746000000.pdf DenseSIFT http://www.vision.caltech.edu/Image_Datasets/Caltech101/cvpr06b_lana.pdf LBP http://www.autex.oxu.lt/publications/pami.02.opm.pdf Sparse SIFT http://www.robots.ox.ac.uk/~vsg/publications/papers/sivic04b.pdf Textron http://www.ics.uci.edu/~fowlkes/papers/mftm-iccv01.pdf
論文ページ: http://cs.brown.edu/~hays/papers/sun.pdf	
プロジェクトページ: http://vision.princeton.edu/projects/2010/SUN/	


ITツールを活用してディスカッション

一人で読むよりも、全員で協力して進めるのが現代流


– クラウド（GoogleDrive/Dropbox）, チャット（Slack/Skype/Line）

 12:45 PM
遅れはせながらReadingListのURLをCVFのものに置換しました。
今年のCVFのページにはarXivのリンクもついていますね、年次の投稿数遷移が取れるかと思いましたが、ICCV2015にはリンクなかったです、残念。

 ☆
ワークショップの動向:
若いワークショップが多い?印象です
> ざっくり数えたところ今回が4回目以下のWSが16件/44件
面白そうなワークショップ
> COMPUTER VISION PROBLEMS IN PLANT PHENOTYPING (CVPPP)
> <https://www.plant-phenotyping.org/CVPPP2017>
> 日本は農業に力を入れてる印象であり、国内学会(View, SSIIなど)だと農業分野の外観検査関連の研究があるので、このWSは今後ねらい目ではないでしょうか

(edited)
 12:04 PM
葉っぱはセグメンテーション&カウントのチャレンジですね。
かなり成熟した感じのある画像認識技術を、アプリケーション寄りにすそ野を広げていきたいという意図が働いているのでしょうか？
非CV屋さんを取り込みたい？

 GANチュートリアル（スライド含む） link: <https://sites.google.com/view/iccv-2017-gans/schedule>

 hirokatsu.kataoka 2:13 PM
1. Mask-RCNN
は前述の通りResNet, Faster RCNNで有名な Kaimingさんの著書です。物体 検出とセマンティックセグメンテーションを同時に解いた方が良い、という知見に基づいています。これ以降、Kaimingさんの成果の速度がゆるくなっているのです、実はこの論文が（検出やセグメンテーションの）完成版で、彼はずでに違うことにフォーカスして 研究しているのでは？と見ています。


2. Kd-network


ICCV 2017 速報の資料作成


スライドを共有

資料をみんなで作り上げていく

- Ver.を上げていくごとに自分と他人の知見を混ぜていく
- 間接的に読んで、議論を重ねるうちに自分にも知識が定着


 10 PM
簡易版テンプレートでspotlightとoralを13本まとめました。そして、今日現時点の10月まとめ資料をppt資料に挿入・変換しました。現在ICCV2017論文まとめは合計75本です。どうぞ、よろしくお願いいたします。こういった資料を入れてV6からV7をアップロードします。


 hirokatsu.kataoka 10:27 PM
uploaded and commented on this file ▼

 **iccv17_update.pptx**
90 kB PowerPoint Presentation

“ 各自のアップデート分のみを更新する資料も準備しました！こちらの該当部分に書き込んで送ってもらえたら更新部分が丸わかりであります。

October 28th, 2017

 1 PM
uploaded and commented on this file ▼

 **iccv17_update_qiu.pptx**
16 MB PowerPoint Presentation

“ 気づきはなかなか書きづらいです。ほぼ論文まとめなので、すみません。

CVPR/ICCV 2017 速報の資料作成

可視化，競争

可視化すると意識して読むようになる

- 週間，月間ランクなど
- 機関総合 (2018/3/1～2018/3/31) 1位：163本, 2位：95本, 3位：87本
- 個人総合 (2018/3/1～2018/3/31) 1位：80本, 2位：68本, 3位：45本

総合集計です (3/1 ~ /31)

――


機関対抗(3/1～3/31)

1位：AIST 163/8 = 20.375

2位：WASEDA 95/5 = 19.000

3位：TDU 87/5 = 17.400

個人ランク(3/1～3/31)

1位：  80

2位：  68

3位：  45

気をつけていること（グループサーベイ）

集団はできる限り仲良くなるように

- リアルだとお茶会/懇親会をする
- リモートだとおしゃべりチャンネルみたいのを作る？

個人へのケアを大事にする

- 活動量が減っている個人と話す（責めずに状況を聞く）
- 活躍したら讃える

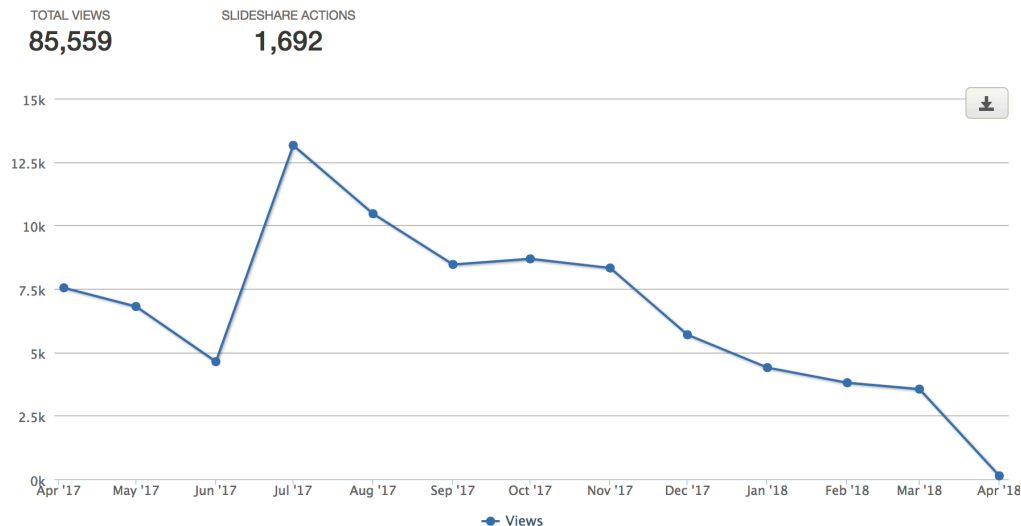
理想は全員が自由に発言できる集団

資料公開のススメ

人目に触れて叩き上げる, プレゼンスを高める

- 評価を見て資料の出来栄えを判断（面白いかどうかくらいはわかる）
- 学会などでリアルに会うと反応をもらえる

論文が通った時だけ宣伝, ではなく
普段やっていることでプレゼンスを上げる



Top content

Name	Views
CVPR 2017 速報	22,526
ICCV 2017 速報	9,486
コンピュータビジョンの今を映す-CVPR 2017 速報より- (夏のトップカンファレンス論文読み会)	4,176
【2017.03】 cvpaper.challenge2017	2,649
CVPR 2016 まとめ v1	2,315

サーベイ法まとめ

サーベイの方法論について、個人/グループという単位で説明

- 速読と精読を組み合わせた個人サーベイ
- 組織的サーベイで早く/確実に知識を作り上げていく

もっと大事なこと

サーベイ（に限らず研究）は楽しんでやるもの！こんな楽しいことができるようになった，分からなかったことを知識として明らかにした，を知る贅沢