frontiers in Robotics and AI

TECHNOLOGY REPORT
published: 11 January 2016
doi: 10.3389/frobt.2015.00036

A Taxonomy of Deep Convolutional Neural Nets for Computer Vision

*Suraj Srinivas, Ravi Kiran Sarvadevabhatla, Konda Reddy Mopuri, Nikita Prabhu, Srinivas S. S. Kruthiventi and R. Venkatesh Babu**

Video Analytics Laboratory, Department of Computational and Data Sciences, Indian Institute of Science, Bangalore, India

Traditional architectures for solving computer vision problems and the degree of success they enjoyed have been heavily reliant on hand-crafted features. However, of late, deep learning techniques have offered a compelling alternative – that of automatically learning problem-specific features. With this new paradigm, every problem in computer vision is now being re-examined from a deep learning perspective. Therefore, it has become important to understand what kind of deep networks are suitable for a given problem. Although general surveys of this fast-moving paradigm (i.e., deep-networks) exist, a survey specific to computer vision is missing. We specifically consider one form of deep networks widely used in computer vision – convolutional neural networks (CNNs). We start with "AlexNet" as our base CNN and then examine the broad variations proposed over time to suit different applications. We hope that our recipe-style survey will serve as a guide, particularly for novice practitioners intending to use deep-learning techniques for computer vision.

Keywords: deep learning, convolutional neural networks, object classification, recurrent neural networks, supervised learning

OPEN ACCESS

Edited by:
Peter M. Hall,
University of Bath, UK

Reviewed by:
Ignacio Arganda-Carreras,
Basque Country University, Spain
Nicolas Pugeault,
University of Surrey, UK

***Correspondence:**
R. Venkatesh Babu
venky@serc.iisc.ernet.in

Specialty section:
This article was submitted to Vision Systems Theory, Tools and Applications, a section of the journal Frontiers in Robotics and AI

Received: *09 October 2015*
Accepted: *10 December 2015*
Published: *11 January 2016*

Citation:
Srinivas S, Sarvadevabhatla RK, Mopuri KR, Prabhu N, Kruthiventi SSS and Babu RV (2016) A Taxonomy of Deep Convolutional Neural Nets for Computer Vision. Front. Robot. AI 2:36. doi: 10.3389/frobt.2015.00036

1. INTRODUCTION

Computer vision problems, such as image classification and object detection, have traditionally been approached using hand-engineered features, such as SIFT by Lowe (2004) and HoG by Dalal and Triggs (2005). Representations based on the Bag-of-visual-words descriptor (see Yang et al., 2007), in particular, enjoyed success in image classification. These were usually followed by learning algorithms like support vector machines (SVMs). As a result, the performance of these algorithms relied crucially on the features used. This meant that progress in computer vision was based on hand-engineering better sets of features. With time, these features started becoming more and more complex – resulting in a difficulty with coming up better, more complex features. From the perspective of the computer vision practitioner, there were two steps to be followed: feature design and learning algorithm design, both of which were largely independent.

Meanwhile, some researchers in the machine learning community had been working on learning models that incorporated learning of features from raw images. These models typically consisted of multiple layers of non-linearity. This property was considered to be very important – and this lead to the development of the first deep learning models. Early examples, such as Restricted Boltzmann Machines (Hinton, 2002), Deep Belief Networks (Hinton et al., 2006) and Stacked Autoencoders (Vincent et al., 2010), showed promise on small datasets. The primary idea behind these works was to leverage the vast amount of unlabeled data to train models. This was called the "unsupervised pre-training" stage. It was believed that these "pre-trained" models would serve as a good initialization for further supervised tasks, such as image classification. Efforts to scale these algorithms on larger

datasets culminated in 2012 during the ILSVRC competition (see Russakovsky et al., 2015), which involved, among other things – the task of classifying an image into 1 of 1000 categories. For the first time, a convolutional neural network (CNN)-based deep learnt model by Krizhevsky et al. (2012) brought down the error rate on that task by half, beating traditional hand-engineered approaches. Surprisingly, this could be achieved by performing end-to-end supervised training, without the need for unsupervised pre-training. Over the next couple of years, "Imagenet classification using deep neural networks" by Krizhevsky et al. (2012) became one of the most influential papers in computer vision. Convolutional neural networks, a particular form of deep learning models, have since been widely adopted by the vision community. In particular, the network trained by Alex Krizhevsky, popularly called "AlexNet" has been used and modified for various vision problems. Hence, in this article, we primarily discuss CNNs, as they are more relevant to the vision community. With the plethora of deep convolutional networks that exist for solving different tasks, we feel the time is right to summarize CNNs for a survey. This article can also serve as a guide for beginning practitioners in deep learning/computer vision.

The paper is organized as follows. We first develop the general principles behind CNNs (see Introduction to Convolutional Neural Networks), and then discuss various modifications to suit different problems (see CNN Flavors). Finally, we discuss some open problems (see Open Problems) and directions for further research.

2. INTRODUCTION TO CONVOLUTIONAL NEURAL NETWORKS

The idea of a convolutional neural network (CNN) is not new. This model had been shown to work well for hand-written digit recognition by LeCun et al. (1998). However, due to the inability of these networks to scale to much larger images, they slowly fell out of favor. This was largely due to memory and hardware constraints, and the unavailability of large amounts of training data. With increase in computational power thanks to wide availability of GPUs, and the introduction of large-scale datasets, such as the ImageNet (see Russakovsky et al., 2015) and the MIT Places dataset (see Zhou et al., 2014), it was possible to train larger, more complex models. This was first shown by the popular *AlexNet* model that was discussed earlier. This largely kick-started the usage of deep networks in computer vision.

2.1. Building Blocks of CNNs

In this section, we shall look at the basic building blocks of CNNs in general. This assumes that the reader is familiar with traditional neural networks, which we shall call "fully connected layers" in this article. **Figure 1** shows a representation of the weights in the *AlexNet* model. While the first five layers are convolutional, the last three are fully connected layers.

2.1.1. Why Convolutions?

Using traditional neural networks for real-world image classification is impractical for the following reason: Consider a 2D image of size 200×200 for which we would have 40,000 input nodes. If the hidden layer has 20,000 nodes, the size of the matrix of input weights would be $40{,}000 \times 20{,}000 = 800$ Million. This is just for the first layer – as we increase the number of layers, this number increases even more rapidly. Besides, vectorizing an image completely ignores the complex 2D spatial structure of the image. How do we build a system that overcomes both these disadvantages?

One way is to use 2D convolutions instead of matrix multiplications. Learning a set of convolutional filters (each of 11×11, say) is much more tractable than learning a large matrix (40000×20000). 2D convolutions also naturally take the 2D structure of images into account. Alternately, convolutions can also be thought of as regular neural networks with two constraints (see Bishop, 2006).

- Local connectivity: this comes from the fact that we use a convolutional filter with dimensions much smaller than the image it operates on. This contrasts with the *global* connectivity paradigm typically relevant to vectorized images.
- Weight sharing: this comes from the fact that we perform convolutions, i.e., we apply the same filter across the image. This means that we use the same *local* filters on many locations in the image. In other words, the weights between all these filters are shared.

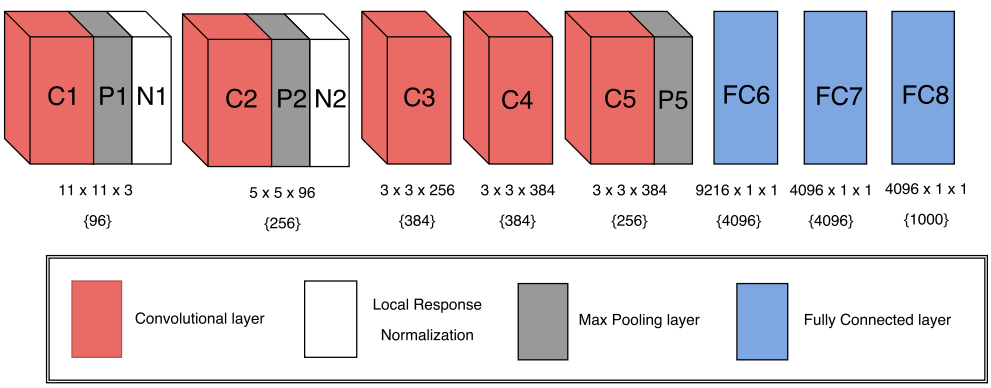

FIGURE 1 | An illustration of the weights in the AlexNet model. Note that after every layer, there is an implicit ReLU non-linearity. The number inside curly braces represents the number of filters with dimensions mentioned above it.

There is also evidence from visual neuroscience for similar computations within the human brain. Hubel and Wiesel (1962) found two types of cells in the primary visual cortex – the simple cells and the complex cells. The simple cell responded primarily to oriented edges and gratings – which are reminiscent of Gabor filters, a special class of convolutional filters. The complex cells were also sensitive to these edges and grating. However, they exhibited spatial invariance as well. This motivated the Neocognitron model by Fukushima (1980), which proposed the learning of convolutional filters in an artificial neural network. This model is said to have inspired convolutional networks, which are analogous to the simple cells mentioned above.

In practical CNNs, however, the convolution operations are not applied in the traditional sense wherein the filter shifts one position to the right after each multiplication. Instead, it is common to use larger shifts (commonly referred to as stride). This is equivalent to performing image down-sampling after regular convolution.

If we wish to train these networks on RGB images, one would need to learn multiple *multi-channel* filters. In the representation in **Figure 1**, the numbers $11 \times 11 \times 3$, along with {96} below **C1** indicates that there are 96 filters in the first layers, each of spatial dimension of 11×11, with one for each of the three RGB channels.

We note that this paradigm of convolution-like operations (location independent feature-detectors) is not entirely suitable for registered images. As an example, images of faces require different feature-detectors at different spatial locations. To account for this, Taigman et al. (2014) consider only locally connected networks with no weight-sharing. Thus, the choice of layer connectivity depends on the underlying type of problem.

2.1.2. Max-Pooling

The Neocognitron model inspired the modeling of simple cells as convolutions. Continuing in the same vein, the complex cells can be modeled as a max-pooling operation. This operation can be thought of as a *max filter*, where each $n \times n$ region is replaced with its max value. This operation serves the following two purposes:

1. It picks out the highest activation in a local region, thereby providing a small degree of spatial invariance. This is analogous to the operation of complex cells.
2. It reduces the size of the activation for the next layer by a factor of n^2. With a smaller activation size, we need a smaller number of parameters to be learnt in the later layers.

2.1.3. Non-Linearity

Deep networks usually consist of convolutions followed by a non-linear operation after each layer. This is necessary because cascading linear systems (like convolutions) is another linear system. Non-linearities between layers ensure that the model is more expressive than a linear model.

In theory, no non-linearity has more expressive power than any other, as long as they are continuous, bounded, and monotonically increasing (see Hornik, 1991). Traditional feedforward neural networks used the sigmoid $\left(\sigma(x) = \frac{1}{1+e^{-x}}\right)$ or the tanh $\left(\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}\right)$ non-linearities. However, modern convolutional networks use the ReLU ($\mathrm{ReLU}(x) = max(0,x)$) non-linearity. CNNs with this non-linearity have been found to train faster, as shown by Nair and Hinton (2010).

Recently, Maas et al. (2013) introduced a new kind of non-linearity, called the leaky-ReLU. It was defined as Leaky-$\mathrm{ReLU}(x) = max(0,x) + \alpha \min(0,x)$, where α is a pre-determined parameter. He et al. (2015) improved on this by suggesting that the α parameter also be learnt, leading to a much richer model.

2.2. Depth

The Universal Approximation theorem by Hornik (1991) states that a neural network with a single hidden layer is sufficient to model any continuous function. However, Bengio (2009) showed that such networks need an exponentially large number of neurons when compared to a neural network with many hidden layers. Recently, Romero et al. (2014) and Ba and Caruana (2014) explicitly showed that a deeper neural network can be trained to perform much better than a comparatively shallow network.

Although the motivation for creating deeper networks was clear, for a long time, researchers did not have an algorithm that could efficiently train neural networks with more than three layers. With the introduction of greedy layerwise pre-training by Hinton et al. (2006), researchers were able to train much deeper networks. This played a major role in bringing the so-called *Deep Learning* systems into mainstream machine learning. Modern deep networks, such as AlexNet, have seven layers. More recent networks, such as VGGnet, by Simonyan and Zisserman (2014b) and GoogleNet by Szegedy et al. (2014) have 19 and 22 layers, respectively, were shown to perform much better than AlexNet.

2.3. Learning Algorithm

A powerful, expressive model is of no use without an algorithm to learn the model's parameters efficiently. The greedy layerwise pre-training approaches in the pre-AlexNet era attempted to create such an efficient algorithm. However, for computer vision tasks, it turned out that a simpler supervised training procedure was enough to learn a powerful model.

Learning is generally performed by minimization of certain loss functions. Tasks based on classification use the softmax loss function or the sigmoid cross entropy function, while those involving regression use the Euclidean error function. In the example of **Figure 1**, the output of the FC8 layer is trained to represent 1 of 1000 classes of the dataset.

2.3.1. Gradient-Based Optimization

Neural networks are generally trained using the backpropogation algorithm (see Rumelhart et al., 1988), which uses the chain rule to speed up the computation of the gradient for the gradient descent (GD) algorithm. However, for datasets with thousands (or more) of data points, using GD is impractical. In such cases, an approximation called the stochastic gradient descent (SGD) is used. It has been found that training using SGD generalizes much better than training using GD. However, one disadvantage is that SGD is very slow to converge. To counteract this, SGD is typically used with a mini-batch, where the mini-batch typically contains a small number of data-points (~100).

Momentum (see Polyak, 1964) belongs to a family of methods that aim to speed the convergence of SGD. This is largely used in practice to train deep networks, and is often considered as an essential component. Other extensions, such as Adagrad by Duchi et al. (2011), Nesterov's accelerated GD by Nesterov (1983), Adadelta by Zeiler and Fergus (2014) and Adam by Kingma and Ba (2014) are known to work equally well, if not better than vanilla momentum in certain cases. For detailed discussion on how these methods work, the reader is encouraged to read Sutskever et al. (2013).

2.3.2. Dropout

When training a network with a large number of parameters, an effective regularization mechanism is essential to combat overfitting. Usual approaches such as λ_1 or λ_2 regularization on the weights of the neural net have been found to be insufficient in this aspect. Dropout is a powerful regularization method introduced by Hinton et al. (2012), which has been shown to work well for large neural nets. To use dropout, we *randomly* drop neurons with a probability p during training. As a result, only a random subset of neurons are trained in a single iteration of SGD. At test time, we use all neurons, however, we simply multiply the activation of each neuron with p to account for the scaling. Hinton et al. (2012) showed that this procedure was equivalent to training a large ensemble of neural nets with shared parameters, and then using their geometric mean to obtain a single prediction.

Many extensions to dropout such as DropConnect by Wan et al. (2013) and Fast Dropout by Wang and Manning (2013) have been shown to work better in certain cases. Maxout by Goodfellow et al. (2013) is a non-linearity that improves performance of a network that uses dropout.

2.4. Tricks to Increase Performance

While the techniques and components described above are theoretically well-grounded, certain *tricks* are crucial to obtaining *state-of-the-art* performance.

It is well known that machine learning models perform better in the presence of more data. Data augmentation is a process by which some geometric transforms are applied to training data to increase their number. Some examples of commonly used geometric transforms include random cropping, RGB jittering, image flipping, and small rotations. It has been found that using augmented data typically boosts performance by about 3% (see Chatfield et al., 2014).

Also well-known is the fact that an ensemble of models perform better than one. Hence, it is the commonplace to train several CNNs and average their predictions at test time. Using ensembles has been found to typically boost accuracy by 1–2% (see Simonyan and Zisserman, 2014b; Szegedy et al., 2014).

2.5. Putting It All Together: AlexNet

The building blocks discussed above largely describe AlexNet as a whole. As shown in **Figure 1**, only layers 1, 2, and 5 contain max-pooling, while dropout is only applied to the last two fully connected layers as they contain the most number of parameters. Layers 1 and 2 also contain local response normalization, which has not been discussed as Chatfield et al. (2014) showed that its absence does not impact performance.

This network was trained on the ILSVRC 2012 training data, which contained 1.2 million training images belonging to 1000 classes. This was trained on two GPUs over the course of 1 month. The same network can be trained today in little under a week using more powerful GPUs (see Chatfield et al., 2014). The hyper-parameters of the learning algorithms, such as learning rate, momentum, dropout, and weight decay, were hand tuned. It is also interesting to note the trends in the nature of features learnt at different layers. The earlier layers tend to learn gabor-like oriented edges and blob-like features, followed by layers that seem to learn more higher order features like shapes. The very last layers seem to learn semantic attributes, such as eyes or wheels, which are crucial parts in several categories. A method to visualize these was provided by Zeiler (2012).

2.6. Using Pre-Trained CNNs

One of the main reasons for the success of the AlexNet model was that it was possible to directly use the pre-trained model to do various other tasks, which it was not originally intended for. It became remarkably easy to download a learnt model, and then tweak it slightly to suit the application at hand. We describe two such ways to use models in this manner.

2.6.1. Fine-Tuning

Given a model trained for image classification, how does one modify it to perform a different (but related) task? The answer is to just use the trained weights as an initialization and run SGD again for this new task. Typically, one uses a learning rate much lower than what was used for learning the original net. If the new task is very similar to the task of image classification (with similar categories), then one need not re-learn a lot of layers. The earlier layers can be fixed and only the later, more semantic layers need to be re-learnt. However, if the new task is very different, one ought to either re-learn all layers, or learn everything from scratch. The number of layers to re-learn also depends on the number of data points available for training the new task. The more the data, the higher is the number of layers that can be re-learnt. The reader is urged to refer to Yosinski et al. (2014) for more thorough guidelines.

2.6.2. CNN Activations as Features

As remarked earlier, the later layers in AlexNet seem to learn visually semantic attributes. These intermediate representations are crucial in performing 1000-way classification. Since these represent a wide variety of classes, one can use the FC7 activation of an image as a generic feature descriptor. These features have been found to be better than hand-crafted features, such as SIFT or HoG for various computer vision tasks.

Donahue et al. (2013) first introduced the idea of using CNN activations as features and performed tests to determine their suitability for various tasks. Babenko et al. (2014) proposed to use the activations of fully connected layers for image retrieval, which they dubbed "Neural Codes". Razavian et al. (2014) used these activations for various tasks and concluded that off-the-shelf CNN features can serve as a hard-to-beat baseline for many

tasks. Hariharan et al. (2014) used activations *across* layers as a feature. Specifically, they look at the activations produced by single-image pixels across the network and pool them together. They were found to be useful for fine-grained tasks, such as keypoint localization.

2.7. Improving AlexNet

The performance of AlexNet motivated a number of CNN-based approaches, all aimed at a performance improvement over and above that of AlexNet's. Just as AlexNet was the winner for ILSVRC challenge in 2012, a CNN-based net Overfeat by Sermanet et al. (2013a) was the top-performer at ILSVRC-2013. Their key insight was that training a convolutional network to simultaneously classify, locate, and detect objects in images can boost the classification accuracy and the detection and localization accuracy of all tasks. Given its multi-task learning paradigm, we discuss Overfeat when we discuss hybrid CNNs and multi-task learning in Section "Hybrid Learning Methods."

GoogleNet by Szegedy et al. (2014), the top-performer at ILSVRC-2014, established that very deep networks can translate to significant gains in classification performance. Since naively increasing the number of layers results in a large number of parameters, the authors employ a number of "design tricks." One such trick is to have a trivial 1×1 convolutional layer after a regular convolutional layer. This has the net effect of not only reducing the number of parameters but also results in CNNs with more expressive power. This design trick is laid out in better detail in the work of Szegedy et al. (2014) where the authors show that having one or more 1×1 convolutional layers is akin to having a multi-layer perceptron network processing the outputs of the convolutional layer that precedes it. Another trick that the authors utilize is to involve inner layers of the network in the computation of the objective function instead of the typical final softmax layer (as in AlexNet). The authors attribute scale invariance as the reason behind this design decision.

VGG-19 and its variants by Simonyan and Zisserman (2014b) is another example of a high-performing CNN where the deeper-is-better philosophy is applied in the net design. An interesting feature of VGG design is that it forgoes larger sized convolutional filters for stacks of smaller sized filters. These smaller sized filters tend to be chosen so that they contain approximately the same number of parameters as the larger filters they supposedly replace. The net effect of this design decision is efficiency and regularization-like effect on parameters due to the smaller size of the filters involved.

3. CNN FLAVORS

3.1. Region-Based CNNs

Most CNNs trained for image recognition are trained using a dataset of images containing a single object. At test time, even in case of multiple objects, the CNN may still predict a single class. This inherent problem with the design of the CNNs is not restricted to image classification alone. For example, the problem of object detection and localization requires not only classifying the image but also estimating the class and precise location of the object(s) present in the image. Object detection is challenging since we potentially want to detect multiple objects with varying sizes within a single image. It generally requires processing the image patch-wise, looking for the presence of objects. Neural nets have been employed in this way for detecting specific objects like faces in Vaillant et al. (1994) and Rowley et al. (1998) and for pedestrians by Sermanet et al. (2013c).

Meanwhile, detecting a set of object-like regions in a given image – also called region proposals or object proposals – has gained a lot of attention (see Uijlings et al., 2013). These region proposals are class agnostic and reduce the overhead incurred by the traditional exhaustive sliding window approach. These region proposal algorithms operate at low level and output hundreds of object like image patches at multiple scales. In order to employ a classification net toward the task of object localization, image patches of different scales have to be searched one at a time.

Recent work by Girshick et al. (2014) attempt to solve the object localization problem using a set of region proposals. During test time, the method generates around 2000 category independent region proposals using selective search by Uijlings et al. (2013) from the test image. They employ a simple affine image warping to feed each of these proposals to a CNN trained for classification. The CNN then describes each of these regions with a fixed size high-level semantic feature. Finally, a set of category-specific linear SVMs classify each region, as shown in **Figure 2**. This method achieved the best detection results on the PASCAL VOC 2012 dataset. As this method uses image regions followed by a CNN, it is dubbed R-CNN (Region-based CNN).

A series of works adapted the R-CNN approach to extract richer set of features at patch or region level to solve a wide range of target applications in vision. However, CNN representations lack robustness to geometric transformations restricting their usage. Gong et al. (2014a) show empirical evidence that the global CNN features are sensitive to general transformations, such as translation, rotation, and scaling. In their experiments, they report that this inability of global CNN features translates directly into a loss in the classification accuracy. They proposed a simple

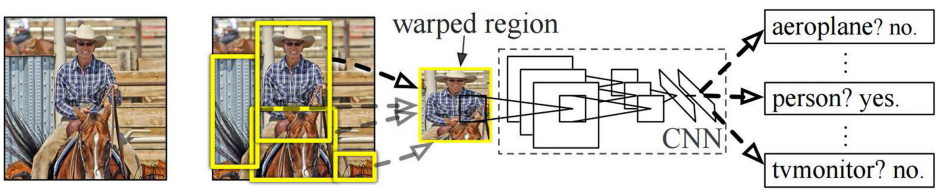

FIGURE 2 | Object detection system of Girshick et al. (2014) using deep features extracted from image regions.

technique to pool the CNN activations extracted from the local image patches. The method extracts image patches in an exhaustive sliding-window manner at different scales and describes each of them using a CNN. The resulting dense CNN features are pooled using VLAD (see Jégou et al., 2011) in order to result in a representation that incorporates spatial as well as semantic information.

Instead of considering the image patches at exhaustive scales and image locations, Mopuri and Babu (2015) utilize the objectness prior to automatically extract the image patches at different scales. They build a more robust image representation by aggregating the individual CNN features from the patches for an image search application.

Wei et al. (2014) extended the capability of a CNN that is trained to output a single label into predicting multiple labels. They consider an arbitrary number of region proposals in an image and share a common CNN across all of them in order to obtain individual predictions. Finally, they employ a simple pooling technique to produce the final multi-label prediction.

3.2. Fully Convolutional Networks

The success of convolutional neural networks in the tasks of image classification (see Krizhevsky et al., 2012; Szegedy et al., 2014) and object detection (see Girshick et al., 2014) has inspired researchers to use deep networks for more challenging recognition problems, such as semantic object segmentation and scene parsing. Unlike image classification, semantic segmentation and scene parsing are problems of structured prediction where every pixel in the image grid needs to be assigned a label of the class to which it belongs (e.g., road, sofa, table, etc.). This problem of per-pixel classification has been traditionally approached by generating region-level (e.g., superpixel) hand crafted features and classifying them using a support vector machine (SVM) into one of the possible classes.

Doing away with these engineered features, Farabet et al. (2013a) used hierarchical learned features from a convolutional neural net for scene parsing. Their approach comprised of densely computing multi-scale CNN features for each pixel and aggregating them over image regions upon which they are classified. However, their method still required the post-processing step of generating over-segmented regions, such as superpixels, for obtaining the final segmentation result. Additionally, the CNNs used for multi-scale feature learning were not very deep with only three convolution layers.

Later, Long et al. (2015) proposed a fully convolutional network architecture for learning per-pixel tasks, such as semantic segmentation, in an end-to-end manner. This is shown in **Figure 3**. Each layer in the fully convolutional net (FullConvNet) performs a location invariant operation i.e., a spatial shift of values in the input to the layer will only result in an equivalent scaled spatial shift in its output while keeping the values nearly intact. This property of translational invariance holds true for the convolutional and max-pool layers that form the major building blocks of a FullConvNet. Furthermore, these layers have an output-centered, fixed-size receptive field on its input blob. These properties of the layers of FullConv Net allow it to retain the spatial structure present in the input image in all of its intermediate and final outputs.

Unlike CNNs used for image classification, a FullConvNet does not contain any densely connected/inner product layers as they are not translation invariant. The restriction on the size of input image to a classification CNN [e.g., 227×227 for AlexNet (Krizhevsky et al., 2012), 224×224 for VGG (Simonyan and Zisserman, 2014b)] is imposed due to the constraint on the input size to its inner product layers. Since a FullConvNet does not have any of these inner product layers, it can essentially operate on input images of any arbitrary size.

During the design of CNN architectures, one has to make a trade-off between the number of channels and the spatial dimensions for the data as it passes through each layer. Generally, the number of channels in the data is made to increase progressively while bringing down its spatial resolution, by introducing stride in the convolution and max-pool layers of the net. This is found to be an effective strategy for generating richer semantic representations in a hierarchical manner. While this method enables the net to recognize complex patterns in the data, it also diminishes the spatial resolution of the data blob progressively after each layer. While this is not a major concern for classification nets that require only a single label for the entire image, this results in per-pixel prediction only at a sub-sampled resolution in case of FullConvNets. For tackling this problem, Long et al. (2015) have proposed a deconvolution layer that brings back the spatial resolution from the sub-sampled output through a learned upsampling operation. This upsampling operation is performed at intermediate layers of various spatial dimensions and are concatenated to obtain pixel-level features at the original resolution.

On the other hand, Chen et al. (2014) adopted a more simplistic approach for maintaining resolution by removing the stride in

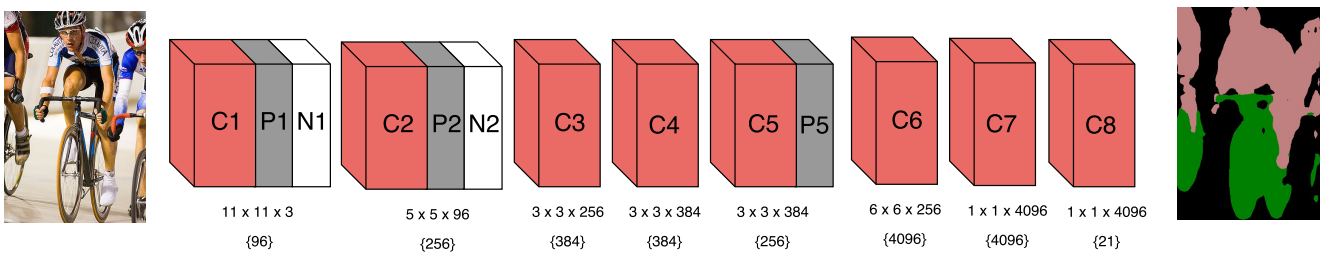

FIGURE 3 | Fully Convolutional Net: AlexNet modified to be fully convolutional for performing semantic object segmentation on PASCAL VOC 2012 dataset with 21 classes.

the layers of FullConvNet, wherever possible. Following this, the FullConvNet predicted output is modeled as a unary term for conditional random field (CRF) constructed over the image grid at its original resolution. With labeling smoothness constraint enforced through pair-wise terms, the per-pixel classification task is modeled as a CRF inference problem. While this post-processing of FullConvNet's coarse labeling using CRF has been shown to be effective for pixel-accurate segmentation, Zheng et al. (2015) have proposed a better approach where the CRF constructed on image is modeled as a recurrent neural network (RNN). By modeling the CRF as an RNN, it can be integrated as a part of any Deep Convolutional Net making the system efficient at both semantic feature extraction and fine-grained structure prediction. This enables the end-to-end training of the entire FullConvNet + RNN system using the stochastic gradient descent (SGD) algorithm to obtain fine pixel-level segmentation.

Visual saliency is another important pixel-level problem considered by researchers. This task involves predicting the salient regions of an image given by human eye fixations. Works by Vig et al. (2014) and Liu et al. (2015) proposed CNN-based approaches for estimating the saliency score for constituent image patches using deep features. By contrast, Kruthiventi et al. (2015) proposed a FullConvNet architecture – DeepFix, which learnt to predict saliency for the entire image in an end-to-end fashion and attained a superior performance. Their network characterized the multi-scale aspects of the image using inception blocks and captured the global context using convolutional layers with large receptive fields. Another work, by Li et al. (2015b), proposed a multi-task FullConvNet architecture – DeepSaliency for joint saliency detection and semantic object segmentation. Their work showed that learning features collaboratively for two related prediction tasks can boost overall performance.

3.3. Multi-Modal Networks

The success of CNNs on standard RGB vision tasks is naturally extended to works on other perception modalities, such as RGB-D and motion information in the videos. Recently, there has been an increasing evidence for the successful adaptation of the CNNs to learn efficient representations from the depth images. Socher et al. (2012) exploited the information from color and depth modalities for addressing the problem of classification. In their approach, a single layer of CNN extracts low-level features from both the RGB and depth images separately. These low-level features from each modality are given to a set of RNNs for embedding into a lower dimension. Concatenation of the resulting features forms the input to the final soft-max layer. The work by Couprie et al. (2013) extended the CNN method of Farabet et al. (2013b) to label the indoor scenes by treating depth information as an additional channel to the existing RGB data. Similarly, Wang et al. (2014a) adapt an unsupervised feature learning approach to scene labeling using RGB-D input with four channels. Gupta et al. (2014) proposed an encoding for the depth images that allows CNNs to learn stronger features than from the depth image alone. They encode depth image into three channels at each pixel: horizontal disparity, height above ground, and the angle the pixel's local surface normal makes with the inferred gravity direction. Their approach for object detection and segmentation processes RGB and the encoded depth channels separately. The learned features are fused by concatenating and further fed into a SVM.

Similarly, one can think of extending these works for video representation and understanding. When compared to still images, videos provide important additional information in the form of motion. However, majority of the early works that attempted to extend CNNs for video, fed the networks with raw frames. This makes for a much difficult learning problem. Jhuang et al. (2007) proposed a biologically inspired model for action recognition in videos with a predefined set of spatio-temporal filters in the initial layer. Combined with a similar but spatial HMAX (Hierarchical model and X) model, Kuehne et al. (2011) proposed spatial and temporal recognition streams. Ji et al. (2010) addressed an end-to-end learning of the CNNs for videos for the first time using 3-D convolutions over a bunch of consecutive video frames. A more recent work by Karpathy et al. (2014) propose a set of techniques to fuse the appearance information present from a stack of consecutive frames in a video. However, they report that the net that processes individual frames performs on par with the net that operates on a stack of frames. This might suggest that the learnt spatio-temporal filters are not suitable to capture the motion patterns efficiently.

A more suitable CNN model to represent videos is proposed in a contemporaneous work by Simonyan and Zisserman (2014a), which is called two-stream network approach. Though the model in Kuehne et al. (2011) is also a two-stream model, the main difference is that the streams are shallow and implemented with hand-crafted models. The reason for the success of this approach is the natural ability of the videos to be separated into spatial and temporal components. The spatial component in the form of frames captures the appearance information, such as the objects, present in the video. The temporal component in the form of motion (optical flow) across the frames captures the movement of the objects. These optical flow estimates can be obtained either from classical approaches (see Baker and Matthews, 2004) or deep-learnt approaches (see Weinzaepfel et al., 2013).

This approach models the recognition system dividing into two parallel streams as depicted in **Figure 4**. Each is implemented by a dedicated deep CNN, whose predictions are later fused. The net for the spatial stream is similar to the image recognition CNN and processes one frame at a time. However, the temporal stream takes the stacked optical flow of a bunch of consecutive frames as input and predicts the action. Both the nets are trained separately with the corresponding input. An alternative motion representation using the trajectory information similar to Wang and Schmid (2013) is also observed to perform similar to optical flow.

The most recent methods that followed Simonyan and Zisserman (2014a) have similar two-stream architecture. However, their contribution is to find the most active spatio-temporal volume for the efficient video representation. Inspired from the recent progress in the object detection in images, Gkioxari and Malik (2015) built action models from shape and motion cues. They start from the image proposals and select the motion salient subset of them and extract spatio-temporal features to represent the video using the CNNs.

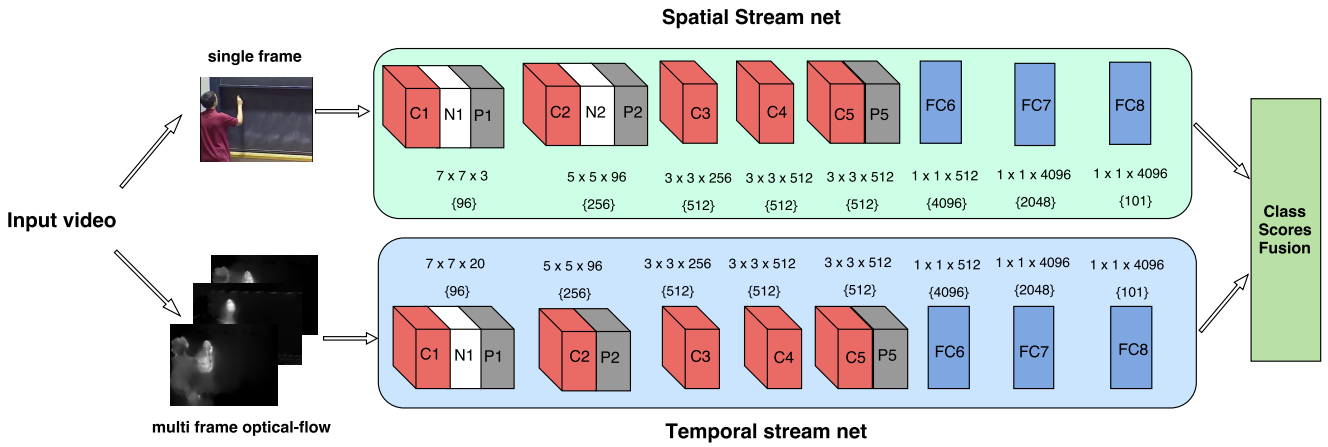

FIGURE 4 | Two-stream architecture for video classification from Simonyan and Zisserman (2014a).

Wang et al. (2015a) employ deep CNNs to learn discriminative feature maps and conduct trajectory constrained pooling to summarize into an effective video descriptor. The two streams operate in parallel extracting local deep features for the volumes centered around the trajectories.

In general, these multi-modal CNNs can be modified and extended to suit any other kind of modality, such as audio, text to complement the image data leading to a better representation of image content.

3.4. CNNs with RNNs

While CNNs have made remarkable progress in various tasks, they are not very suitable for learning sequences. Learning such patterns requires memory of previous states and feedback mechanisms that are not present in CNNs. RNNs are neural nets with at least one feedback connection. This looping structure enables the RNN to have an internal memory and to learn temporal patterns in data.

Figure 5 shows the unrolled version of a simple RNN applied to a toy example of sequence addition. The problem is defined as follows: let a_t be a positive number, corresponding to the input at time t. The output at time t is given by

$$S_t = \sum_{i=1}^{t} a_i.$$

We consider a very simple RNN with just one hidden layer. The RNN can be described by equations below.

$$h_{t+1} = f_h(W_{ih} \times a_t + W_{hh} \times h_t)$$
$$S_{t+1} = f_o(W_{ho} \times h_{t+1})$$

where W_{ih}, W_{hh}, W_{ho} are learned weights and f_h and f_o are non-linearities. For the toy problem considered above, the weights learned would result in $W_{ih} = W_{hh} = W_{ho} = 1$. Let us consider the non-linearity to be ReLu. The equations would then become,

$$h_{t+1} = ReLu(a_t + h_t)$$
$$S_{t+1} = ReLu(h_{t+1}).$$

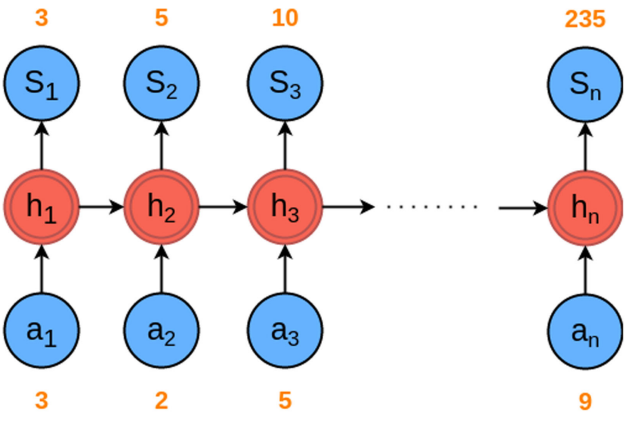

FIGURE 5 | Toy RNN example: problem of sequence addition. The inputs and outputs are shown in blue. The red cells correspond to the hidden units. An unrolled version of the RNN is shown.

Thus, as shown in **Figure 5**, the RNN stores previous inputs in memory and learns to predict the sum of the sequence up to the current timestep t.

As with CNNs, recurrent neural networks have been trained with various back propagation techniques. These conventional methods however, resulted in the *vanishing gradient problem*, i.e., the errors sent backward over the network, either grew very large or vanished leading to problems in convergence. In 1997, Hochreiter and Schmidhuber (1997) introduced LSTM (Long Short Term Memory), which succeeded in overcoming the vanishing gradient problem, by introducing a novel architecture consisting of units called constant error carousels. LSTMs were, thus, able to learn very deep RNNs and successfully remembered important events over long (thousands of steps) durations of time.

Over the next decade, LSTM's became the network of choice for several sequence learning problems, especially in the fields of speech and handwriting recognition (see Graves et al., 2009, 2013). In the sections that follow, we shall discuss applications of RNNs in various computer vision problems.

3.4.1. Action Recognition

Recognizing human actions from videos has long been a pivotal problem in the tasks of video understanding and surveillance. Actions, being events that take place over a finite length of time, are excellent candidates for a joint CNN-RNN model.

In particular, we discuss the model proposed by Donahue et al. (2014). They use *RGB* as well as *optical flow* features to jointly train a variant of Alexnet combined with a single layer of LSTM (256 hidden units). Frames of the video are sampled, passed through the trained network, and classified individually. The final prediction is obtained by averaging across all the frames. A snapshot of this model at time *t* is shown in **Figure 6**.

3.4.2. Image and Video Captioning

Another important component of scene understanding is the textual description of images and videos. Relevant textual description also helps complement image information, as well as form useful queries for retrieval.

RNNs (LSTMs) have long been used for machine translation (see Bahdanau et al., 2014; Cho et al., 2014). This has motivated its use for the purpose of image description. Vinyals et al. (2014) have developed an end-to-end system, by first encoding an image using a CNN and then using the encoded image as an input to a language generating RNN. Karpathy and Fei-Fei (2014) propose a multimodal deep network that aligns various interesting regions of the image, represented using a CNN feature, with associated words. The learned correspondences are then used to train a bi-directional RNN. This model is able not only to generate descriptions for images, but also to localize different segments of the sentence to their corresponding image regions. The multimodal RNN (m-RNN) by Mao et al. (2014) combines the functionalities of the CNN and RNN by introducing a new multimodal layer, after the embedding and recurrent layers of the RNN. Mao et al. (2015) further extend the m-RNN by incorporating a transposed weight sharing strategy, enabling the network to learn novel visual concepts from images.

Venugopalan et al. (2014) move beyond images and obtain a mean-pooled CNN representation for a video. They train an LSTM to use this input to generate a description for the video. They further improve upon this task by developing S2VT (Venugopalan et al., 2015) a stacked LSTM model that accounts for both the RGB as well as flow information available in videos. Pan et al. (2015) use both 2-D and 3-D CNNs to obtain a video embedding. They introduced two types of losses that are used to train both the LSTM and the visual semantic embedding.

3.4.3. Visual Question Answering

Real understanding of an image should enable a system not only to make a statement about it, but also to answer questions related to it. Therefore, answering questions based on visual concepts in an image is the next natural step for machine understanding algorithms. Doing this, however, requires the system to model both the textual question and the image representation, before generating an answer conditioned on both the question and the image.

A combination of CNN and LSTM has proven to be effective in this task too, as evidenced by the work of Malinowski et al. (2015) who train an LSTM layer to accept both the question as well a CNN representation of the image and generate the answer. Gao et al. (2015) use two LSTMs with shared weights along with a CNN for the task. Their experiments are performed on a multilingual dataset containing Chinese questions and answers along with its English translation. Antol et al. (2015) provide a dataset for the task of visual question answering containing both real-world images and abstract scenes.

3.5. Hybrid Learning Methods

3.5.1. Multi-Task Learning

Multi-task learning is essentially a machine learning paradigm wherein the objective is to train the learning system to perform well on multiple tasks. Multi-task learning frameworks tend to exploit shared representations that exist among the tasks to obtain a better generalization performance than counterparts developed for a single task alone.

In CNNs, multi-task learning is realized using different approaches. One class of approaches utilize a multi-task loss

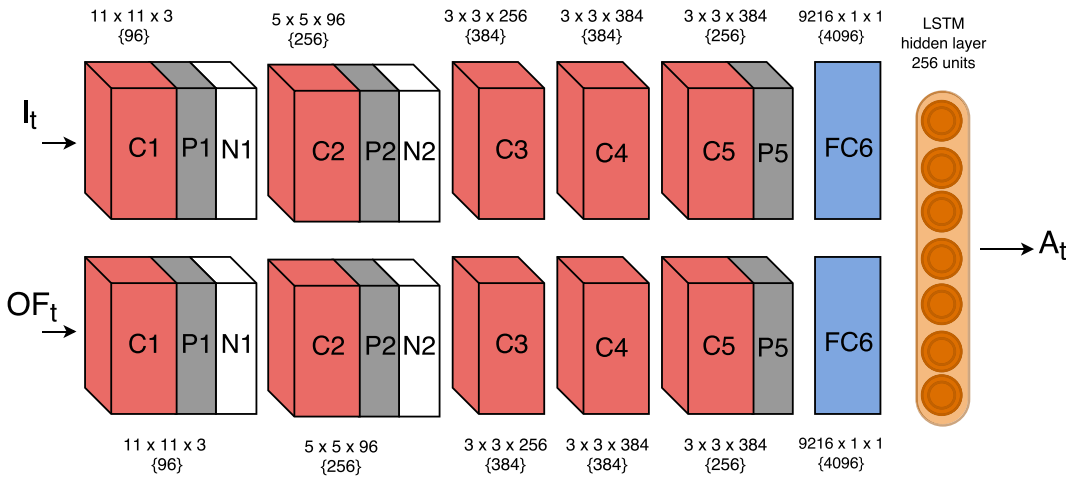

FIGURE 6 | LRCN: a snapshot of the model at time *t*. I_t refers to the frame and OF_t the optical flow at t. The video is classified by averaging the output A_t over all t.

function with hyper-parameters typically regulating the task losses. For example, Girshick (2015) employ a multi-task loss to train their network jointly for classification and bounding-box regression tasks thereby improving performance for object detection. Zhang et al. (2014) propose a facial landmark detection network that adaptively weights auxiliary tasks (e.g., head pose estimation, gender classification, age estimation) to ensure that a task that is deemed not beneficial to accurate landmark detection is prevented from contributing to the network learning. Devries et al. (2014) demonstrate improved performance for facial expression recognition task by designing a CNN for simultaneous landmark localization and facial expression recognition. A hallmark of these approaches is the division of tasks into primary task and auxiliary task(s) wherein the purpose of the latter is typically to improve the performance of the former (see **Figure 7**).

Some approaches tend to have significant portions of the original network modified for multiple tasks. For instance, Sermanet et al. (2013b) replace pre-trained layers of a net originally designed to provide spatial (per-pixel) classification maps with a regression network and fine-tune the resulting net to achieve simultaneous classification, localization, and detection of scene objects.

Another class of multi-task approaches tend to have task-specific sub-networks as a characteristic feature of CNN design. Li et al. (2015a) utilize separate sub-networks for the joint point regression and body part detection tasks. Wang et al. (2015b) adopt a serially stacked design wherein a localization sub-CNN and the original object image are fed into a segmentation sub-CNN to generate its object bounding box and extract its segmentation mask. To solve an unconstrained word image recognition task, Jaderberg et al. (2014a) propose an architecture consisting of a character sequence CNN and an N-gram encoding CNN that act on an input image in parallel and whose outputs are utilized along with a CRF model to recognize the text content present within the image.

3.5.2. Similarity Learning

Apart from classification, CNNs can also be used for tasks, such as metric learning and rank learning. Rather than asking the CNN to identify objects, we can instead ask it to verify whether two images contain the same object or not. In other words, we ask the CNN to learn which images are *similar*, and which are not. Image retrieval is one application where such questions are routinely asked.

Structurally, Siamese networks resemble two-stream networks discussed previously. However, the difference here is that both "streams" have identical weights. Siamese networks consist of two separate (but identical) networks, where two images are fed in as input. Their activations are combined at a later layer, and the output of the network consists of a single number, or a *metric*, which is a notion of distance between the images. Training is done so that images that are considered to be similar have a lower output score than images that are considered different. Bromley et al. (1993) separate introduced the idea of Siamese networks and used it for signature verification. Later on, Chopra et al. (2005) extended it for face verification. Zagoruyko and Komodakis (2015) further extended and generalized this to learning similarity between image patches.

Triplet networks are extensions of siamese networks used for rank learning. Wang et al. (2014b) first used this idea for learning fine-grained image similarity learning.

4. OPEN PROBLEMS

In this section, we briefly mention some open research problems in deep learning, particularly of interest to computer vision. Several of these problems are already being tackled in several works.

- Training CNNs requires tuning of a large number of hyper-parameters, including those involving the model architecture. An automated way of tuning such as that by Snoek et al. (2012) is crucial for practitioners. However, that requires multiple models to be trained, which can be both time consuming and impractical for large networks.
- Nguyen et al. (2014) showed that one can generate artificial images that result in CNNs producing a high confidence false prediction. In a related line of work, Szegedy et al. (2013) showed that natural images can be modified in an imperceptible manner to produce a completely different classification label. Although Goodfellow et al. (2014a) attempted to reduce the effects of such adversarial examples, it remains to be seen whether that can be completely eliminated.

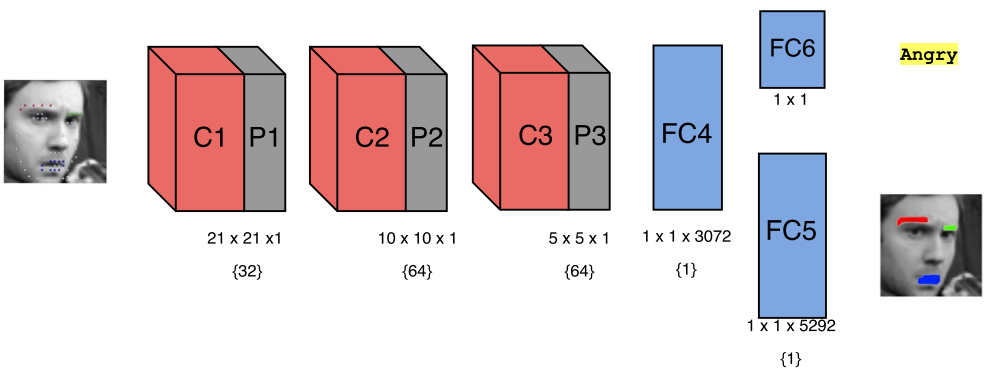

FIGURE 7 | The facial expression recognition system of Devries et al. (2014) that utilizes facial landmark (shown overlaid on the face toward the right of the image) recognition as an auxiliary task that helps improve performance on the main task of expression recognition.

- It is well known (see Gong et al., 2014b) that CNNs are *robust* to small geometric transforms. However, we would like them to be *invariant*. The study of *invariance* for extreme deformation is largely missing.
- Along with using a large number of data points, CNN models are also large and (relatively) slow to evaluate. While there has been a lot of work in reducing number of parameters (see Denil et al., 2013; Collins and Kohli, 2014; Jaderberg et al., 2014b; Hinton et al., 2015; Srinivas and Babu, 2015), it is not clear how to train non-redundant models in the first place.
- CNNs are presently trained in a *one-shot* way. The formulation of an *online* method of training would be desirable for robotics applications.
- Unsupervised learning is one more area where we expect to deploy deep learning models. This would enable us to leverage the massive amounts of unlabeled image data on the web. Classical deep networks, such as autoencoders and restricted Boltzmann machines, were formulated as unsupervised models. While there has been a lot of interesting recent work in the area (see Bengio et al., 2013; Kingma and Welling, 2013; Goodfellow et al., 2014b; Kulkarni et al., 2015), a detailed discussion of these is beyond the scope of this paper.

5. CONCLUDING REMARKS

In this article, we have surveyed the use of deep learning networks – convolutional neural networks in particular – for computer vision. This enabled complicated hand-tuned algorithms being replaced by single monolithic algorithms trained in an end-to-end manner. However, despite our best efforts, it may not be possible to capture the entire gamut of deep learning research – even for computer vision – in this paper. We point the reader to other reviews, specifically those by Bengio (2009), LeCun et al. (2015), and Schmidhuber (2015). These reviews are more geared toward deep learning in general, while ours is more focused on computer vision. We hope that our article will be useful to vision researchers beginning to work in deep learning.

AUTHOR CONTRIBUTIONS

SS contributed to Section 1, part of Section 2, and part of Sections 3 and 4. RS contributed to Section 1, and part of Section 2 and part of Section 3. KM contributed to Sections 3.1 and 3.3. NP contributed to Section 3.4. SK contributed to Section 3.2. VB contributed to overall design and drafting.

REFERENCES

Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., et al. (2015). *VQA: Visual Question Answering*. arXiv preprint arXiv:1505.00468.

Ba, J., and Caruana, R. (2014). "Do deep nets really need to be deep?," in *Advances in Neural Information Processing Systems 27*, eds Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger (Curran Associates, Inc.), 2654–2662.

Babenko, A., Slesarev, A., Chigorin, A., and Lempitsky, V. (2014). "Neural codes for image retrieval," in *Computer Vision – ECCV 2014* (Springer), 584–599.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). *Neural Machine Translation by Jointly Learning to Align and Translate*. arXiv preprint arXiv:1409.0473.

Baker, S., and Matthews, I. (2004). Lucas-kanade 20 years on: a unifying framework. *Int. J. Comput. Vis.* 56, 221–255. doi:10.1023/B:VISI.0000011205.11775.fd

Bengio, Y. (2009). Learning deep architectures for AI. *Found. Trends Mach. Learn.* 2, 1–127. doi:10.1561/2200000006

Bengio, Y., Thibodeau-Laufer, E., Alain, G., and Yosinski, J. (2013). *Deep Generative Stochastic Networks Trainable by Backprop*. arXiv preprint arXiv:1306.1091.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ: Springer-Verlag New York, Inc.

Bromley, J., Bentz, J. W., Bottou, L., Guyon, I., LeCun, Y., Moore, C., et al. (1993). Signature verification using a siamese time delay neural network. *Int. J. Pattern Recogn. Artif. Intell.* 7, 669–688. doi:10.1142/S0218001493000339

Chatfield, K., Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). "Return of the devil in the details: delving deep into convolutional nets," in *Proceedings of the British Machine Vision Conference* (Nottingham: BMVA Press).

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2014). *Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs*. arXiv preprint arXiv:1412.7062.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., et al. (2014). *Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation*. arXiv preprint arXiv:1406.1078.

Chopra, S., Hadsell, R., and LeCun, Y. (2005). "Learning a Similarity metric discriminatively, with application to face verification," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005*, Vol. 1 (IEEE), 539–546.

Collins, M. D., and Kohli, P. (2014). *Memory Bounded Deep Convolutional Networks*. arXiv preprint arXiv:1412.1442.

Couprie, C., Farabet, C., Najman, L., and LeCun, Y. (2013). *Indoor Semantic Segmentation Using Depth Information*. CoRR, abs/1301.3572.

Dalal, N., and Triggs, B. (2005). "Histograms of oriented gradients for human detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005*, Vol. 1 (IEEE), 886–893.

Denil, M., Shakibi, B., Dinh, L., Ranzato, M. A., and de Freitas, N. (2013). "Predicting parameters in deep learning," in *Advances in Neural Information Processing Systems 26*, eds C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger (Curran Associates, Inc.), 2148–2156.

Devries, T., Biswaranjan, K., and Taylor, G. W. (2014). "Multi-task learning of facial landmarks and expression," in *Canadian Conference on Computer and Robot Vision (CRV), 2014*, 98–103.

Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., et al. (2014). *Long-Term Recurrent Convolutional Networks for Visual Recognition and Description*. arXiv preprint arXiv:1411.4389.

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., et al. (2013). *Decaf: A Deep Convolutional Activation Feature for Generic Visual Recognition*. arXiv preprint arXiv:1310.1531.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.* 12, 2121–2159.

Farabet, C., Couprie, C., Najman, L., and LeCun, Y. (2013a). Learning hierarchical features for scene labeling. *IEEE Trans. Pattern Anal. Mach. Intell.* 35, 1915–1929. doi:10.1109/TPAMI.2012.231

Farabet, C., Couprie, C., Najman, L., and LeCun, Y. (2013b). Learning hierarchical features for scene labeling. *IEEE Trans. Pattern Anal. Mach. Intell.* 35, 1915–1929. doi:10.1109/TPAMI.2012.231

Fukushima, K. (1980). Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol. Cybern.* 36, 193–202. doi:10.1007/BF00344251

Gao, H., Mao, J., Zhou, J., Huang, Z., Wang, L., and Xu, W. (2015). Are you talking to a machine? Dataset and methods for multilingual image question answering. arXiv preprint arXiv:1505.05612.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). "Rich feature hierarchies for accurate object detection and semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014* (IEEE), 580–587.

Girshick, R. B. (2015). *Fast R-CNN*. CoRR, abs/1504.08083.

Gkioxari, G., and Malik, J. (2015). "Finding action tubes," in *CVPR*.

Gong, Y., Wang, L., Guo, R., and Lazebnik, S. (2014a). "Multi-scale orderless pooling of deep convolutional activation features," in *Computer Vision–ECCV 2014* (Springer International Publishing), 392–407.

Gong, Y., Wang, L., Guo, R., and Lazebnik, S. (2014b). "Multi-scale orderless pooling of deep convolutional activation features," in *Computer Vision–ECCV 2014* (Springer International Publishing), 392–407.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014a). *Explaining and Harnessing Adversarial Examples*. arXiv preprint arXiv:1412.6572.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., et al. (2014b). "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, eds Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger (Curran Associates, Inc.), 2672–2680.

Goodfellow, I. J., Warde-Farley, D., Mirza, M., Courville, A., and Bengio, Y. (2013). *Maxout Networks*. arXiv preprint arXiv:1302.4389.

Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., and Schmidhuber, J. (2009). A novel connectionist system for unconstrained handwriting recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 31, 855–868. doi:10.1109/TPAMI.2008.137

Graves, A., Mohamed, A.-R., and Hinton, G. (2013). "Speech recognition with deep recurrent neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2013* (IEEE), 6645–6649.

Gupta, S., Girshick, R., Arbelaez, P., and Malik, J. (2014). *Learning Rich Features from RGB-D Images for Object Detection and Segmentation*.

Hariharan, B., Arbeláez, P., Girshick, R., and Malik, J. (2014). *Hypercolumns for Object Segmentation and Fine-Grained Localization*. arXiv preprint arXiv:1411.5752.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). *Delving Deep into Rectifiers: Surpassing Human-Level Performance on Imagenet Classification*. arXiv preprint arXiv:1502.01852.

Hinton, G., Vinyals, O., and Dean, J. (2015). *Distilling the Knowledge in a Neural Network*. arXiv preprint arXiv:1503.02531.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Comput.* 14, 1771–1800. doi:10.1162/089976602760128018

Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Comput.* 18, 1527–1554. doi:10.1162/neco.2006.18.7.1527

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012). *Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors*. arXiv preprint arXiv:1207.0580.

Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.* 9, 1735–1780. doi:10.1162/neco.1997.9.8.1735

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Netw.* 4, 251–257. doi:10.1016/0893-6080(91)90009-T

Hubel, D. H., and Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *J. Physiol.* 160, 106. doi:10.1113/jphysiol.1962.sp006837

Jaderberg, M., Simonyan, K., Vedaldi, A., and Zisserman, A. (2014a). *Deep Structured Output Learning for Unconstrained Text Recognition*. CoRR, abs/1412.5903.

Jaderberg, M., Vedaldi, A., and Zisserman, A. (2014b). *Speeding up Convolutional Neural Networks with Low Rank Expansions*. arXiv preprint arXiv:1405.3866.

Jégou, H., Perronnin, F., Douze, M., Sánchez, J., Pérez, P., and Schmid, C. (2011). Aggregating local image descriptors into compact codes. *IEEE Trans. Pattern Anal. Mach. Intell.*

Jhuang, H., Serre, T., Wolf, L., and Poggio, T. (2007). "A biologically inspired system for action recognition," in *IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007*, 1–8.

Ji, S., Xu, W., Yang, M., and Yu, K. (2010). 3D convolutional neural networks for human action recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 35, 221–231. doi:10.1109/TPAMI.2012.59

Karpathy, A., and Fei-Fei, L. (2014). *Deep Visual-Semantic Alignments for Generating Image Descriptions*. arXiv preprint arXiv:1412.2306.

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. (2014). "Large-scale video classification with convolutional neural networks," in *Computer Vision and Pattern Recognition (CVPR)* (Columbus, OH: IEEE), 1725–1732.

Kingma, D., and Ba, J. (2014). *Adam: A Method for Stochastic Optimization*. arXiv preprint arXiv:1412.6980.

Kingma, D. P., and Welling, M. (2013). *Auto-Encoding Variational Bayes*. arXiv preprint arXiv:1312.6114.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, eds F. Pereira, C. Burges, L. Bottou, and K. Weinberger (Curran Associates, Inc.), 1097–1105.

Kruthiventi, S. S., Ayush, K., and Babu, R. V. (2015). *Deepfix: A Fully Convolutional Neural Network for Predicting Human Eye Fixations*. arXiv preprint arXiv:1510.02927.

Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., and Serre, T. (2011). "HMDB: a large video database for human motion recognition," in *IEEE International Conference on Computer Vision (ICCV), 2011* (IEEE), 2556–2563.

Kulkarni, T. D., Whitney, W., Kohli, P., and Tenenbaum, J. B. (2015). *Deep Convolutional Inverse Graphics Network*. arXiv preprint arXiv:1503.03167.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature* 521, 436–444. doi:10.1038/nature14539

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 2278–2324. doi:10.1109/5.726791

Li, S., Liu, Z.-Q., and Chan, A. (2015a). Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network. *Int. J. Comput. Vis.* 113, 19–36. doi:10.1007/s11263-014-0767-8

Li, X., Zhao, L., Wei, L., Yang, M., Wu, F., Zhuang, Y., et al. (2015b). *Deepsaliency: Multi-Task Deep Neural Network Model for Salient Object Detection*. arXiv preprint arXiv:1510.05484.

Liu, N., Han, J., Zhang, D., Wen, S., and Liu, T. (2015). *Predicting Eye Fixations Using Convolutional Neural Networks*.

Long, J., Shelhamer, E., and Darrell, T. (2015). "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3431–3440.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* 60, 91–110. doi:10.1023/B:VISI.0000029664.99615.94

Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, Vol. 30.

Malinowski, M., Rohrbach, M., and Fritz, M. (2015). *Ask Your Neurons: A Neural-Based Approach to Answering Questions about Images*. arXiv preprint arXiv:1505.01121.

Mao, J., Xu, W., Yang, Y., Wang, J., Huang, Z., and Yuille, A. (2015). *Learning like a Child: Fast Novel Visual Concept Learning from Sentence Descriptions of Images*. arXiv preprint arXiv:1504.06692.

Mao, J., Xu, W., Yang, Y., Wang, J., and Yuille, A. (2014). *Deep Captioning with Multimodal Recurrent Neural Networks (m-RNN)*. arXiv preprint arXiv:1412.6632.

Mopuri, K. R., and Babu, R. V. (2015). "Object level deep feature pooling for compact image representation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 62–70.

Nair, V., and Hinton, G. E. (2010). "Rectified linear units improve restricted Boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 807–814.

Nesterov, Y. (1983). "A method of solving a convex programming problem with convergence rate o (1/k2)," in *Soviet Mathematics Doklady*, Vol. 27, 372–376.

Nguyen, A., Yosinski, J., and Clune, J. (2014). *Deep neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images*. arXiv preprint arXiv:1412.1897.

Pan, Y., Mei, T., Yao, T., Li, H., and Rui, Y. (2015). *Jointly Modeling Embedding and Translation to Bridge Video and Language*. arXiv preprint arXiv:1505.01861.

Polyak, B. T. (1964). Some methods of speeding up the convergence of iteration methods. *USSR Comput. Math. Math. Phys.* 4, 1–17. doi:10.1016/0041-5553(64)90137-5

Razavian, A. S., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). "CNN features off-the-shelf: an astounding baseline for recognition," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2014* (IEEE), 512–519.

Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. (2014). *Fitnets: Hints for Thin Deep Nets*. arXiv preprint arXiv:1412.6550.

Rowley, H. A., Baluja, S., and Kanade, T. (1998). Neural network-based face detection. *IEEE Trans. Pattern Anal. Mach. Intell.* 20, 23–38. doi:10.1109/34.655647

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1988). Learning representations by back-propagating errors. *Cogn. Model.* 5, 3.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.* 115, 211–252. doi:10.1007/s11263-015-0816-y

Schmidhuber, J. (2015). Deep learning in neural networks: an overview. *Neural Netw.* 61, 85–117. doi:10.1016/j.neunet.2014.09.003

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. (2013a). *Overfeat: Integrated Recognition, Localization and Detection using Convolutional Networks*. arXiv preprint arXiv:1312.6229.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. (2013b). *Overfeat: Integrated Recognition, Localization and Detection using Convolutional Networks*. CoRR, abs/1312.6229.

Sermanet, P., Kavukcuoglu, K., Chintala, S., and LeCun, Y. (2013c). "Pedestrian detection with unsupervised multi-stage feature learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013* (IEEE), 3626–3633.

Simonyan, K., and Zisserman, A. (2014a). "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems 27*, eds Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger 568–576.

Simonyan, K., and Zisserman, A. (2014b). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv preprint arXiv:1409.1556.

Snoek, J., Larochelle, H., and Adams, R. P. (2012). "Practical bayesian optimization of machine learning algorithms," in *Advances in Neural Information Processing Systems,* 25, eds F. Pereira, C. Burges, L. Bottou, and K. Weinberger (Curran Associates, Inc.), 2951–2959.

Socher, R., Huval, B., Bath, B., Manning, C. D., and Ng, A. Y. (2012). "Convolutional-recursive deep learning for 3d object classification," in *Advances in Neural Information Processing Systems 25*, eds F. Pereira, C. Burges, L. Bottou, and K. Weinberger, 656–664.

Srinivas, S., and Babu, R. V. (2015). "Data-free parameter pruning for deep neural networks," in *Proceedings of the British Machine Vision Conference (BMVC)*, eds M. W. J. Xianghua Xie and G. K. L. Tam (BMVA Press), 31.1–31.12.

Sutskever, I., Martens, J., Dahl, G., and Hinton, G. (2013). "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*. 1139–1147.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., et al. (2014). *Going Deeper with Convolutions*. arXiv preprint arXiv:1409.4842.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., et al. (2013). *Intriguing Properties of Neural Networks*. arXiv preprint arXiv:1312.6199.

Taigman, Y., Yang, M., Ranzato, M., and Wolf, L. (2014). "Deepface: closing the gap to human-level performance in face verification," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014* (IEEE), 1701–1708.

Uijlings, J., van de Sande, K., Gevers, T., and Smeulders, A. (2013). Selective search for object recognition. *Int. J. of Comput. Vision* doi:10.1007/s11263-013-0620-5

Vaillant, R., Monrocq, C., and Le Cun, Y. (1994). Original approach for the localisation of objects in images. *IEE Proc. Vision Image Signal Process.* 141, 245–250. doi:10.1049/ip-vis:19941301

Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., and Saenko, K. (2015). *Sequence to Sequence – Video to Text*. arXiv preprint arXiv:1505.00487.

Venugopalan, S., Xu, H., Donahue, J., Rohrbach, M., Mooney, R., and Saenko, K. (2014). *Translating Videos to Natural Language Using Deep Recurrent Neural Networks*. arXiv preprint arXiv:1412.4729.

Vig, E., Dorr, M., and Cox, D. (2014). "Large-scale optimization of hierarchical features for saliency prediction in natural images," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014* (IEEE), 2798–2805.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P.-A. (2010). Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.* 11, 3371–3408.

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. (2014). *Show and Tell: A Neural Image Caption Generator*. arXiv preprint arXiv:1411.4555.

Wan, L., Zeiler, M., Zhang, S., Cun, Y. L., and Fergus, R. (2013). "Regularization of neural networks using dropconnect," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 1058–1066.

Wang, A., Lu, J., Wang, G., Cai, J., and Cham, T.-J. (2014a). "Multi-modal unsupervised feature learning for Rgb-D scene labeling," in *Computer Vision – ECCV 2014*, volume 8693 of *Lecture Notes in Computer Science*, eds D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars (Springer), 453–467.

Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., et al. (2014b). "Learning fine-grained image similarity with deep ranking," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014* (IEEE), 1386–1393.

Wang, H., and Schmid, C. (2013). Action recognition with improved trajectories. In *IEEE International Conference on Computer Vision*, Sydney.

Wang, L., Qiao, Y., and Tang, X. (2015a). "Action recognition with trajectory-pooled deep-convolutional descriptors," in *CVPR*, 4305–4314.

Wang, X., Zhang, L., Lin, L., Liang, Z., and Zuo, W. (2015b). Deep joint task learning for generic object extraction. CoRR, abs/1502.00743.

Wang, S., and Manning, C. (2013). "Fast dropout training," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 118–126.

Wei, Y., Xia, W., Huang, J., Ni, B., Dong, J., Zhao, Y., et al. (2014). *CNN: Single-Label to Multi-Label*. CoRR, abs/1406.5726.

Weinzaepfel, P., Revaud, J., Harchaoui, Z., and Schmid, C. (2013). "Deepflow: Large displacement optical flow with deep matching," in *IEEE International Conference on Computer Vision (ICCV), 2013* (IEEE), 1385–1392.

Yang, J., Jiang, Y.-G., Hauptmann, A. G., and Ngo, C.-W. (2007). "Evaluating bag-of-visual-words representations in scene classification," in *Proceedings of the International Workshop on Multimedia Information Retrieval* (ACM), 197–206.

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). "How transferable are features in deep neural networks?," in *Advances in Neural Information Processing Systems*, 27, eds Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger (Curran Associates, Inc.), 3320–3328.

Zagoruyko, S., and Komodakis, N. (2015). *Learning to Compare Image Patches via Convolutional Neural Networks*. arXiv preprint arXiv:1504.03641.

Zeiler, M., and Fergus, R. (2014). "Visualizing and understanding convolutional networks," in *Computer Vision – ECCV 2014*, volume 8689 of *Lecture Notes in Computer Science*, eds D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars (Springer International Publishing), 818–833. doi:10.1007/978-3-319-10590-1_53

Zeiler, M. D. (2012). *Adadelta: An Adaptive Learning Rate Method*. arXiv preprint arXiv:1212.5701.

Zhang, Z., Luo, P., Loy, C. C., and Tang, X. (2014). *Learning and Transferring Multi-Task Deep Representation for Face Alignment*. CoRR, abs/1408.3967.

Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., et al. (2015). *Conditional Random Fields as Recurrent Neural Networks*. arXiv preprint arXiv:1502.03240.

Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., and Oliva, A. (2014). "Learning deep features for scene recognition using places database," in *Advances in Neural Information Processing Systems 27*, eds Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger (Curran Associates, Inc.), 487–495.

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.