UNIVERSITAT POLITÈCNICA DE CATALUNYA
UNIVERSITAT DE BARCELONA
UNIVERSITAT ROVIRA I VIRGILI

Master in Artificial Intelligence

Master of Science Thesis

MY MASTER THESIS TITLE

Hadi Keivan Ekbatani

FACULTAT D'INFORMÀTICA DE BARCELONA (FIB)
FACULTAT DE MATEMÀTIQUES (UB)
ESCOLA TÈCNICA SUPERIOR D'ENGINYERIA (URV)

Supervisor:
Oriol Pujol Vila
Department of analysis
and applied Mathematics,
Universitat de Barcelona (UB)

Co-supervisor:
Santiago Segui Mesquida
Department of analysis
and applied Mathematics,
Universitat de Barcelona (UB)

February 01, 2016

Acknowledgments

I would like to sincerely thank my supervisors Oriol Pujol and Santi Segui for their support, guidance and mentorship. I greatly appreciate their demanding and inquisitive scientific attitude, while keeping always a calm and positive mindset. They are, in my mind, an inspiring example of what university professors should be.

Furthermore I would also like to dedicate this master thesis to my beloved parents and sister for their unsparing supports. My gratitude knows no bounds.

Another special acknowledgment must be made to my friends in the master: Denis, Jeroni, Philipp, Pablo, Lorenzo, Iosu, Ferran and many others for the endless studying hours, morning coffees after crunching the brutal assignments all night long, valuable and constructive discussions which allowed me carry out this program.

Abstract

NOT YET

Contents

List of Figures

1 Introduction

Learning to count is an important "educational/developmental milestone" — hmm, it's smth different. maybe concept or ability educational/developmental milestone which constitutes the most fundamental idea of mathematics. In Computer Vision, the counting problem is the estimation of the number of objects in a still image or video frame. Learning to count visual objects is a new approach towards dealing with detecting objects in the images and video, which has been recently proffered means offered, do you mean that? proffered in the literaturesuch as paper1, paper2,.... It arises in many real-world applications, including cell counting in microscopic images, monitoring crowds in surveillance systems, and performing wildlife census or counting the number of trees in an aerial image of a forest[30]citation missing.

1.1 Motivations

1.2 Objectives

1.3 Contributions of the Research

This thesis contains the following contributions:

1. It proposes the problem of object representation as an indirect learning problem casted as learning to count strategy. The devised algorithm is capable of counting the number of pedestrians in the image that does not depend on object detection or feature tracking. The model is also privacy-preserving in a sense that it can be implemented with hardware that does not produce a visual record of the individuals in the scene.

2. It provides a synthetically generated and automatically labeled dataset of pedestrians using unlabeled University of California San Diego(UCSD) pedestrian dataset used in [33], to train a counting deep convolutional neural network which is adequate for apprehending the underlying representations of the learned features. To this end, we describe a counting problem for MNIST dataset to demonstrate the capability of the internal representation of the network for classifying digits with no direct supervising while training.

3. The proposed model is able to count the number of people in the real and unseen dataset using the features learned by training the network on synthetic training set. To our knowledge, this is the first crowd counting system trained by synthetic data that successfully operates continuously on real data.

4. Along with the validation of our proposal in the following ways:

- First, we learn to count even hand-written digits using MNIST dataset.
- Second, we validate the system quantitatively on a large synthetic dataset of pedestrian, containing 100,000 images with maximum 30 pedestrians in each image.
- Last but not least, we count the number of pedestrians in the manually labeled dataset of 3375 images provided by[Chan and Vasconcelos, 2013].

1.4 Organization

2 Background and Definitions

In this section, we go over the preliminarily concepts that help understand the contributions of this work. We start by looking at the family of methods to which Deep Convolutional Neural Networks belong, followed by a more detailed look at the method in question better name this 'method in question', explaining the some hyper-parameters incorporated for optimizing the proposed model.i think the explanation of your method should be done in a diff section

2.1 Deep Learning

One of the central challenges of Artificial Intelligence (AI) is solving the tasks that are easy for people to perform but hard for them to describe formally – problems that we solve intuitively, that feel automatic, like recognizing spoken words or faces in images. 'one approach to that challenge' maybe The solution is to allow computers to learn from experience and understand the world in terms of a hierarchy of concepts, where each concept is defined in terms of its relation to simpler concepts. This hierarchy of concepts allows the computer to learn complex notions by building them out of simpler ones. If we draw a graph showing how these concepts are built on top of each other, it would be a deep graph with many layers. For this reason, we call this approach *Deep Learning*[16].

Modern deep learning provides a very powerful framework for supervised learning. By adding more layers and more units within a layer, a deep network can represent functions of increasing complexity. Most tasks that consist of mapping an input vector to an output vector, and that are easy for a person to perform quickly, can be accomplished via deep learning, given sufficiently large models and datasets of labeled training examples. Other tasks, that can not be described as associating one vector to another, or that are difficult enough such that a person would require time to think and reflect in order to accomplish the task, remain beyond the scope of deep learning for now[16].

In other words, Deep Learning is a new area of Machine Learning research, which has been introduced with the objective of moving ML closer to one of its original goals: Artificial Intelligence. Deep Learning is about learning multiple levels of representation and abstraction that help to make sense of data such as images, sound and text[45].

2.2 Deep Neural Networks

there is a package in latex – don't remember how it's called – that lets you define acronynims and reuse them across the text with an automatic list of acronims being generated – ask pablo A standard neural network (NN) consists of many simple, connected processors called neurons, each producing a sequence of real-valued activations. Shallow NN-like models with few such stages have been around for many decades if not centuries[42]. However, theoretical results strongly suggest that in order to learn the kind of complicated functions that can represent high-level abstractions (e.g. in vision, language, and other AI-level tasks), one needs deep architectures. Deep Neural Networks are composed of multiple levels of non-linear operations, such as those present in neural nets with many hidden layers or in complicated propositional formulate re-using many sub-formulate[1]'propositional formulate re-using many sub-formulate' – i dont really understand that.

2.2.1 Back Propagation

Backward Propagation of errors (BP) was the main advance in the 1980's that led to an explosion of interest in NNs. BP is one of the most commonly used methods for training NNs. The idea behind BP is that it repeatedly adjusts the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the network and the desired one. As a result of the weight adjustments, internal *hidden* units come to represent important features of the task domain, and the regularities in the task are captured by the interactions of these units[52].

Specifically, BP computes how fast the error changes as we adjust a hidden activity by using error derivatives with respect to hidden activities'what are hidden activities' – it comes up out of the blue. Since each hidden activity can have a notable effect on many output units and consequently on the error, a combination of these effects must be considered. This aggregation is done efficiently which allows us to compute error derivatives for all the hidden units quickly at the same time. Computing the error derivatives for the hidden activities, it would be easy to get the error derivatives for the weights going into a hidden unit which is the key to be able to learn efficiently.

2.2.2 Weight Sharing

Transition missing: smth like 'another itegral part part/building block/technique' of deep learning is 'weight sharing' Weight sharing refers to having several connections controlled by a single parameter (weight). Weight sharing can be interpreted as imposing equality constraints among the connection strengths. An interesting feature of weight sharing is that it can be implemented with very little computational overhead[25]. The weight sharing technique has an interesting side effect of reducing the number of free parameters, thereby the capacity of the machine and improving its generalization ability[27]. how is weight sharing relevant in this research. for instance, 'we will later try to modify weight sharing / capitalize on it to build a more efficient model'

2.3 Convolutional Neural Networks

Convolutional Neural Networks are a specialized kind of neural network for processing data that has a known grid-like topology such as image data which can be thought of as a 2D grid of pixels. CNN are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers[16]. Essentially, CNNs combine three architectural ideas to ensure some degree of shift and distortion invariance of local receptive fields, shared weights (or weight replication), and, sometimes, spatial or temporal sub-sampling[27] what are 'local receptive fields', 'spacial/temporal subsampling'. The following components compose the main body of any CNN architecture:

2.3.1 Convolutional layer

Each unit of a convolutional layer receives inputs from a set of units located in a small neighborhood in the previous layer. With local receptive fields, neurons can extract elementary visual features such as oriented edges, end-points and corners. These features are then combined by the higher layers[27]. In addition, elementary feature detectors that are useful on one part of the image are likely to be useful across the entire image. This knowledge can be applied by forcing a set of units, whose receptive fields are located at different places on the image, to have identical weight vectors[52]. The outputs of such a set of neurons constitutes a *feature map.* At each position, different types of units in various feature maps compute different types of features. A sequential implementation of this, for each feature map, would be to scan the input image with a single neuron that has a local receptive field, and to store the states of this neuron at corresponding locations in the feature map[27].

Units in a feature map are constrained to perform the same operation on different parts ot the image. A convolutional layer is usually composed of several feature maps (with different weight vectors), so that multiple features can be extracted at each location.

2.3.2 Pooling/Sub-sampling layer

Once a feature is detected, its' exact position becomes less important as long as its' approximate position relative to other features is preserved. Furthermore, as the dimensionality of applying a filter is equal to the input dimensionality, we would not be gaining any translation invariance with these additional filters, we would be stuck doing pixel-wise analysis on increasingly abstract features. In order to solve this problem, a *subsampling* layer is introduced.

Subsampling, or down-sampling, refers to reducing the overall size of a signal. In many cases, such as audio compression for music files, subsampling is done simply for size reduction [46]. But in the domain of 2D filter outputs, subsampling can also be thought of as reducing the sensitivity of the output to shifts and distortions. One of the most applied subsampling methods used in [26], is known as 'max pooling'. This involves splitting up the matrix of filter outputs into small non-overlapping grids (the larger the grid, the greater the signal reduction), and taking the maximum value in each grid as the value in the reduced matrix. By applying such a max pooling layer in between convolutional layers, we can increase spatial abstractness as we raise feature abstractness[46].

2.3.3 Activation functions

To go from one layer to the next, a set of units compute a weighted sum of their inputs from the previous layer and pass the result through a non-linear activation function[23]. There are many possible choices for the non-linear activation functions in a multi-layered network, and the choice of activation functions for the hidden units may often be different from that for the output units. This is a consequence of the fact the hidden and output units perform different roles[2].

At present, the most popular non-linear function is the Rectified Linear Units (ReLU), which is simply the half-wave rectifier $f(z) = max(z, 0)$. In the past decades, neural nets used smoother non-linearities, such as $tanh(z)$ or $1/(1 + exp(-z))$, but ReLU typically learns much faster in networks with many layers, allowing training of a deep supervised network without unsupervised pre-training[23].

The rectifier activation function allows a network to easily obtain sparse representations. For example, after uniform initialization of the weights, around 50% of hidden units continuous output values are real zeros, and this fraction can easily increase with sparsity-including regularization. Apart from being more biologically plausible, sparsity also leads to mathematical advantages. On the other hand, one may hypothesize that the hard saturation at 0 may hurt optimization by blocking gradient back-propagation. However, experimental results done byGlorot et al. suggest that hard zeros can actually help supervised training[15].

2.3.4 Local Response Normalization

ReLUs have the desirable property that they do not require input normalization to prevent them from saturating. If at least some training examples produce a positive input to a ReLU, learning will happen in that neuron. However, we still find that the following local normalization(LRN) scheme aids generalization. This sort of response normalization implements a form of lateral inhibition wtf is that? inspired by the type found in real neurons, creating competition for big activities amongst neuron outputs computed using different kernels[22].

This scheme bears some resemblance to the local contrast normalization scheme proposed by Jarrett et al. in [19] without mean activity subtraction 'mean activity subtraction' – ?? which has led to error rate reduction in [22] and [18].

2.3.5 Fully connected/Inner product layer

Finally, after several convolutional and max pooling layers, the high-level reasoning in the neural network is done via *fully connected layers*(IP). A fully connected layer takes all neurons in the previous layer (be it fully connected, pooling, or convolutional) and connects it to every single neuron it has. Fully connected layers are not spatially located anymore (you can visualize them as one-dimensional), so there can be no convolutional layers after a fully connected layer.

2.4 Model Optimization

this really does not belong to the background section In this section we briefly describe some optimization methods along with definition of hyper-parameters used in our model.

2.4.1 Stochastic Gradient Descent

It has often been proposed to minimize the *empirical risk* (training set performance measure). For more detailed description, see [48]) using *gradient descent*(GD)[3]. The standard gradient descent algorithm updates the parameters θ of the objective *J(θ)*

$$\theta = \theta - \alpha \bigtriangledown_\theta E[(J(\theta)] \tag{2.1}$$

you need to clarify what each variable stands for in this equation where the expectation in the above equation is approximated by evaluating the cost and gradient over the full training set (Empirical Risk Minimization (ERM)). Stochastic Gradient Descent (SGD) simply does away the expectation in the update and computes the gradient of the parameters using only a single or a few training examples. The new update is given by, are you sure a comma is needed before the equasion?

$$\theta = \theta - \alpha \bigtriangledown_\theta J(\theta; x^{(i)}, y^{(i)}) \tag{2.2}$$

with a pair $(x^{(i)}, y^{(i)})$ from the training set[36].

Generally, each parameter update in SGD is computed with respect to a few training examples or a mini-batch as opposed to a single example. The reasons for this are twofold[36]:

1. The variance in the parameter update is reduced, potentially leading to a more stable convergence.

2. It allows the computation to take advantage of highly optimized matrix operations that should be used in a well vectorized computation of the cost and gradient. A typical mini-batch size is 256, although the optimal size of the mini-batch can vary for different applications and architectures.

3. One final but important point regarding SGD is the order in which we present the data to the algorithm. If the data is given in some meaningful order, this can bias the gradient and lead to poor convergence. Generally, a good method to avoid this is to randomly shuffle the data prior to each epoch of training.

2.4.2 Weight Decay

As a part of BP algorithm and a subset of regularization methods, *weight decay* adds a penalty term to the error function by multiplying weights to a factor slightly less than 1 after each update.

It has been observed in numerical simulations that a weight decay can improve generalization in a feed-forward neural network. It is proven that a weight decay has two effects in a linear network. Firstly, it suppresses any irrelevant components of the weight vector by choosing the smallest vector that solves the learning problem. Secondly, if the size is chosen right, a weight decay can suppress some of the effects of static noise on the targets, which improves generalization significantly[35].

2.4.3 Momentum

The *momentum* method introduced by [Polyak, 1964] is a first-order optimization method for accelerating gradient descent that accumulates a velocity vector in directions of persistent reduction in the objective across iterations. Given an objective function $f(\theta)$ to be minimized, momentum is given by:

$$\nu_{t+1} = \mu\nu_t - \varepsilon\bigtriangledown \tag{2.3}$$

$$\theta_{t+1} = \theta_t + \nu_{t+1} \tag{2.4}$$

where $\varepsilon > 0$ is the learning rate, $\mu \in [0, 1]$ is the momentum coefficient, and $\bigtriangledown f(\theta_t)$ is the gradient at θ_t [44].

For example, if the objective has a form of a long shallow ravine leading to the optimum and steep walls on the sides, standard SGD will tend to oscillate across the narrow ravine since the negative gradient will point down one of the steep sides rather than along the ravine towards the optimumcitation for that?. The objectives of deep architectures have this form near local optima and thus standard SGD can lead to very slow convergence particularly after the initial steep gains. Momentum is one method for pushing the objective more quickly along the shallow ravine[36].

3 State of the art review

Counting the number of an object of interest in an image can be approached from two different perspectives, either training an object detector, or training an object counter[43]. In the field of object detection, numerous works have been previously proposed[38, 8, 41, 10, 21, 34, 50]. Most of these research works follow a taxonomy which consists of three paradigms underneath to count the objects:

1. Object detection, which are based on boosting appearance and motion features[51, 50], Bayesian model-based segmentation[55], integrated top-down and bottom-up processing[29, 37][6].

2. Visual feature trajectory clustering. This paradigm counts objects by identifying and tracking visuals over a time period. Feature trajectories with coherent motion are then clustered and the number of clusters is the estimate of the number of moving people[40, 4][6].

3. feature-based regression. These methods usually work by first, subtracting the background, second, measuring various features of the foreground pixels such as total area[38, 10], edge count[8, 41], or texture [34]; and finally estimating the crowd density or crowd count by a regression function, e.g. linear[38, 10], piece-wise linear [41], or neural networks[8, 41].

   In recent years, feature-based regression has also been applied to outdoor scenes. For example, [21] applies neural networks to the histograms of foreground segment areas and edge orientations.[11] estimates the number of people in each foreground segment by matching its shape to a database containing the silhouettes of possible people configurations, but is only applicable when the number of people in each segment is small (empirically, less than 6)[6].

By reason of the fact that almost all the above algorithms detect the whole objects in an image (e.g. whole pedestrians), these methods have moderate performance in very noisy or crowded images with significant occlusion, Wu and Nevatia 53, Lin et al. 32, introduced methods to address this issue. Wu and Nevatia 53 proposed *edgelet features(an edgelet is a short segment of line or curve)* as new type of silhouette oriented features to deal with the problem of detecting individuals in crowded still images. Respectively in [32], Lin et al. used *Accumulated Mosaic Image Difference(AMID)* method to extract crowd areas having irregular motion.

As a similar line of work in the course of object counting and more specifically crowd counting, in [40, 4, 28], different object tracking approaches were taken to detect and count moving objects in the scene. However, the deployment of these vision surveillance technologies are invariably met with skepticism by society at large, given the perception that they could be used to infringe on the individuals' privacy rights. While a number of methods that do not require explicit detection or tracking have been previously proposed[38, 8, 41, 10, 21, 34, 11], they have not fully established the viability of the privacy-preserving approach[6]. The tension of privacy-preserving is common in all areas of data-mining[47, 49].

In order to tackle privacy preserving issue, Chan et al. 6 presented a novel approach with no explicit object segmentation or tracking to estimate the number of people moving in each direction(towards and away from camera) in a privacy-preserving manner. An outline of the crowd counting system appears in figure 3.1:

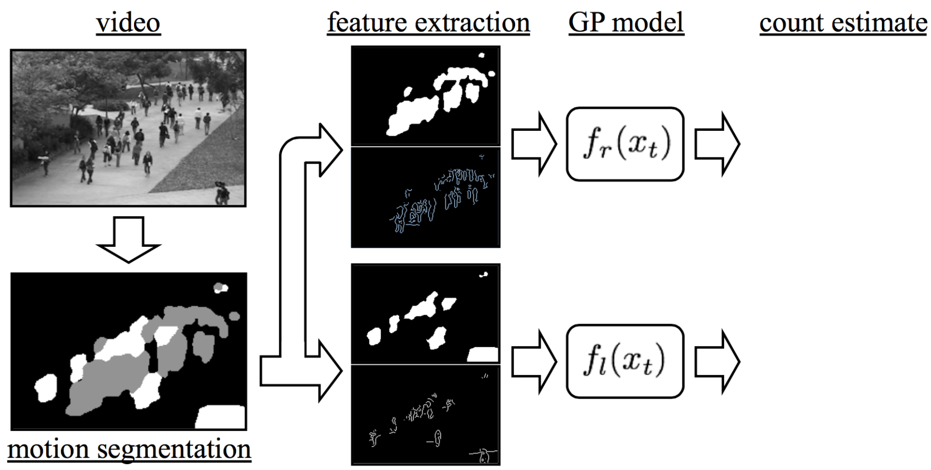

Figure 3.1: Crowd counting system: the scene is segmented into crowds with different motions. Normalized features that account for perspective are extracted from each segment, and the crowd count for each segment is estimated with a Gaussian process[6].

Chan et al. used a mixture of *dynamic textures*[12, 7] to divide the video frames into regions containing moving pedestrians in different directions. When adopting mixture of dynamic textures, the video is represented as collection of spatio-temporal patches which are modeled as independent samples from mixture of dynamic models[12]. The mixture model is learned through Expectation-Maximization(EM) algorithm[7]. Video locations are then scanned sequentially, a patch is extracted at each location, and assigned to the mixture component of largest posterior probability. The location is declared to belong to the segmentation region associated with that component[6]. The resulting segmentations of their work are illustrated in figure 3.2:

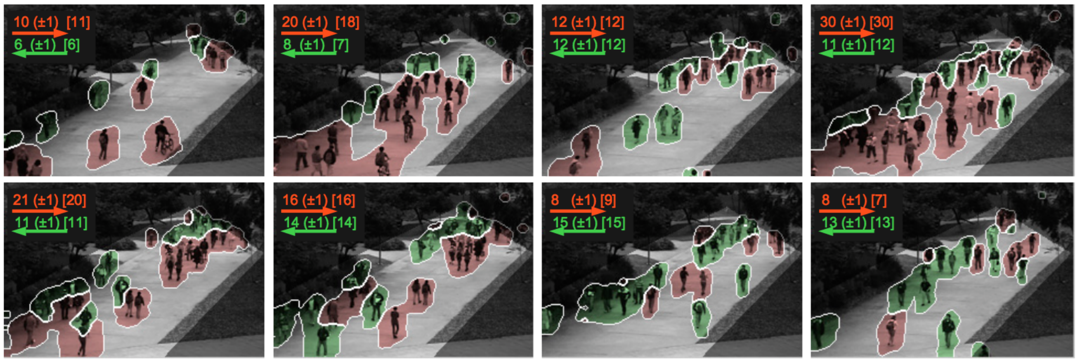

Figure 3.2: Crowd counting results: The red and green segments are the "away" and "towards" crowds. The estimated crowd count for each segment is in the top-left, with the (rounded standard-deviation of the GP) and the [ground-truth]. The Region Of Interest (the area in the walkway in which the pedestrians are counted and labeled) is also highlighted[6].

we must provide the system with a large set of object examples, properly and almost in all cases manually labeled and localized in a way that represent most of the possible views and appearances of the object. The result is a sophisticated object classifier based on manually-crafted features[50, 51].

In the latter case, we only need to provide the number of object instances for each image sample and the result is typically a regressor[31].

Recently, with the success of CNNs in different vision tasks, object detection systems based on deep CNN have made groundbreaking advances on several object detection problems[54, 13, 14, 17, 13] which suggests the use of this technique to learn to count objects. Several advantages can be foreseen from this application, being the most important that of learning image features from samples instead of hand-crafting highly specialized image features that are dependent on the object class[43]. Moreover, CNN have shown their capacity of knowledge transfer for a number of tasks or the ability of simultaneously performing different tasks even when trained for only one [56].

Following this line of work, Seguí et al. in [43] proposed a novel approach for counting objects' representations using deep object features. In their work, objects' features are learned by a deep counting convolutional neural network and are used to understand the underlying representation. To this end, they define a counting problem for even digits using *MNIST* data and demonstrate that the internal representation of the network is able to classify digits in spite of the fact that no direct supervision was provided for them during training. Moreover, they present preliminary results about a deep network that is able to count the number of pedestrians in a scene[43]. Figure 3.3 illustrates their proposal at a glance in the case of representing hand-written digits:

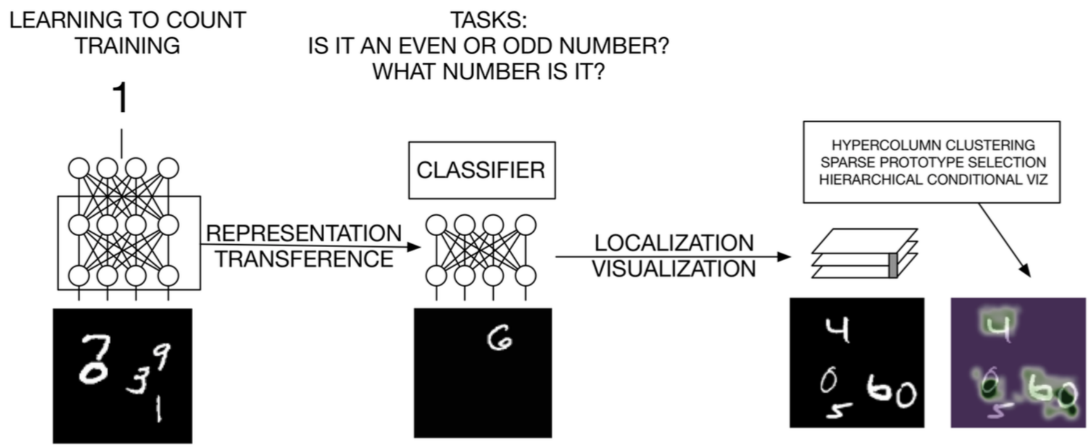

Figure 3.3: Learning to count hand-written digits problem in which the features of a CNN that has been trained to count digits can be readily used for more specific classification problems and even to localize digits in an image[43].

In [43], the main hypothesis is that the number of occurrence of objects in an image provide strong presentational information due to their possible discriminate appearance for a feature learning process to exploit. In order to verify this hypothesis, for both experiments, they considered networks of two or more convolutional layers (since CNNs instinctively handle feature learning[24]) consisting of convolutional filters, ReLU non-linearities, max-pooling layers and normalization layer, followed by one or more fully connected layers (regarding the impressive classification performance on different benchmark problems[22, 20, 9])[43].

References

[1] Bengio, Y. (2009). Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127.

[2] Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford university press.

[3] Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer.

[4] Brostow, G. J. and Cipolla, R. (2006). Unsupervised bayesian detection of independent motion in crowds. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 594–601. IEEE.

[5] Chan, A. and Vasconcelos, N. (2013). Ground truth annotations for ucsd dataset. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

[6] Chan, A. B., Liang, Z.-S. J., and Vasconcelos, N. (2008). Privacy preserving crowd monitoring: Counting people without people models or tracking. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–7. IEEE.

[7] Chan, A. B. and Vasconcelos, N. (2008). Modeling, clustering, and segmenting video with mixtures of dynamic textures. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(5):909–926.

[8] Cho, S.-Y., Chow, T. W., and Leung, C.-T. (1999). A neural-based crowd estimation by hybrid global learning algorithm. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 29(4):535–541.

[9] Ciresan, D. C., Meier, U., Masci, J., Maria Gambardella, L., and Schmidhuber, J. (2011). Flexible, high performance convolutional neural networks for image classification. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, page 1237.

[10] Davies, A. C., Yin, J. H., and Velastin, S. A. (1995). Crowd monitoring using image processing. *Electronics & Communication Engineering Journal*, 7(1):37–47.

[11] Dong, L., Parameswaran, V., Ramesh, V., and Zoghlami, I. (2007). Fast crowd segmentation using shape indexing. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE.

[12] Doretto, G., Chiuso, A., Wu, Y. N., and Soatto, S. (2003). Dynamic textures. *International Journal of Computer Vision*, 51(2):91–109.

[13] Erhan, D., Szegedy, C., Toshev, A., and Anguelov, D. (2014). Scalable object detection using deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2154.

[14] Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587.

[15] Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 315–323.

[16] Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep learning. Book in preparation for MIT Press.

[17] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 37(9):1904–1916.

[18] 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*.

[19] Jarrett, K., Kavukcuoglu, K., Ranzato, M., and LeCun, Y. (2009). What is the best multi-stage architecture for object recognition? In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2146–2153. IEEE.

[20] Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. (2014). Large-scale video classification with convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[21] Kong, D., Gray, D., and Tao, H. (2005). Counting pedestrians in crowds using viewpoint invariant training. In *BMVC*. Citeseer.

[22] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

[23] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

[24] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989a). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551.

[25] LeCun, Y. et al. (1989b). Generalization and network design strategies. *Connections in Perspective. North-Holland, Amsterdam*, pages 143–55.

[26] LeCun, Y., Jackel, L., Bottou, L., Brunot, A., Cortes, C., et al. (1995). Comparison of learning algorithms for handwritten digit recognition. In *Comparison of learning algorithms for handwritten digit recognition*.

[27] LeCun, Y., Kavukcuoglu, K., Farabet, C., et al. (2010). Convolutional networks and applications in vision. In *ISCAS*, pages 253–256.

[28] Leibe, B., Schindler, K., and Van Gool, L. (2007). Coupled detection and trajectory estimation for multi-object tracking. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE.

[29] Leibe, B., Seemann, E., and Schiele, B. (2005). Pedestrian detection in crowded scenes. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 878–885. IEEE.

[30] Lempitsky, V. and Zisserman, A. (2010a). learn. In Lafferty, J., Williams, C., Shawe-Taylor, J., Zemel, R., and Culotta, A., editors, *Advances in Neural Information Processing Systems 23*, pages 1324–1332. Curran Associates, Inc.

[31] Lempitsky, V. and Zisserman, A. (2010b). Learning to count objects in images. In *Advances in Neural Information Processing Systems*, pages 1324–1332.

[32] Lin, S.-F., Chen, J.-Y., and Chao, H.-X. (2001). Estimation of number of people in crowded scenes using perspective transformation. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 31(6):645–654.

[33] Mahadevan, V., Li, W., Bhalodia, V., and Vasconcelos, N. (2010). Anomaly detection in crowded scenes. *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), San Francisco, CA, 2010.*

[34] Marana, A., Costa, L. d. F., Lotufo, R., and Velastin, S. (1998). On the efficacy of texture analysis for crowd monitoring. In *Computer Graphics, Image Processing, and Vision, 1998. Proceedings. SIBGRAPI'98. International Symposium on*, pages 354–361. IEEE.

[35] Moody, J., Hanson, S., Krogh, A., and Hertz, J. A. (1995). A simple weight decay can improve generalization. *Advances in neural information processing systems*, 4:950–957.

[36] Ng, A. and colleagues (2013). http://ufldl.stanford.edu/tutorial/supervised/ optimization-stochasticgradientdescent/.

[37] Oliva, A., Torralba, A., Castelhano, M. S., and Henderson, J. M. (2003). Top-down control of visual attention in object detection. In *Image processing, 2003. icip 2003. proceedings. 2003 international conference on*, volume 1, pages I–253. IEEE.

[38] Paragios, N. and Ramesh, V. (2001). A mrf-based approach for real-time subway monitoring. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–1034. IEEE.

[39] Polyak, B. T. (1964). Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17.

[40] Rabaud, V. and Belongie, S. (2006). Counting crowded moving objects. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 705–711. IEEE.

[41] Regazzoni, C. S. and Tesei, A. (1996). Distributed data fusion for real-time crowding estimation. *Signal Processing*, 53(1):47–63.

[42] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117.

[43] Seguí, S., Pujol, O., and Vitria, J. (2015). Learning to count with deep object features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 90–96.

[44] 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)*, pages 1139–1147.

[45] Tutorial, D. L. (2014). Lisa lab. *University of Montreal*.

[46] University, S. (2013). http://white.stanford.edu/teach/index.php/an_introduction_to_convolutional_neural_networks.

[47] Vaidya, J., Clifton, C. W., and Zhu, Y. M. (2006). *Privacy preserving data mining*, volume 19. Springer Science & Business Media.

[48] Vapnik, V. N. and Vapnik, V. (1998). *Statistical learning theory*, volume 1. Wiley New York.

[49] Verykios, V. S., Bertino, E., Fovino, I. N., Provenza, L. P., Saygin, Y., and Theodoridis, Y. (2004). State-of-the-art in privacy preserving data mining. *ACM Sigmod Record*, 33(1):50–57.

[50] Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *International journal of computer vision*, 57(2):137–154.

[51] Viola, P., Jones, M. J., and Snow, D. (2005). Detecting pedestrians using patterns of motion and appearance. *International Journal of Computer Vision*, 63(2):153–161.

[52] Williams, D. R. G. H. R. and Hinton, G. (1986). Learning representations by back-propagating errors. *Nature*, 323:533–536.

[53] Wu, B. and Nevatia, R. (2005). Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 90–97. IEEE.

[54] Zhang, Y., Sohn, K., Villegas, R., Pan, G., and Lee, H. (2015). Improving object detection with deep convolutional networks via bayesian optimization and structured prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 249–258.

[55] Zhao, T. and Nevatia, R. (2003). Bayesian human segmentation in crowded situations. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–459. IEEE.

[56] 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*, pages 487–495.