Computational Models of Cognition: Part VI: Deep Learning (DL)

Alireza Dehbozorgi

20 min read · Just now

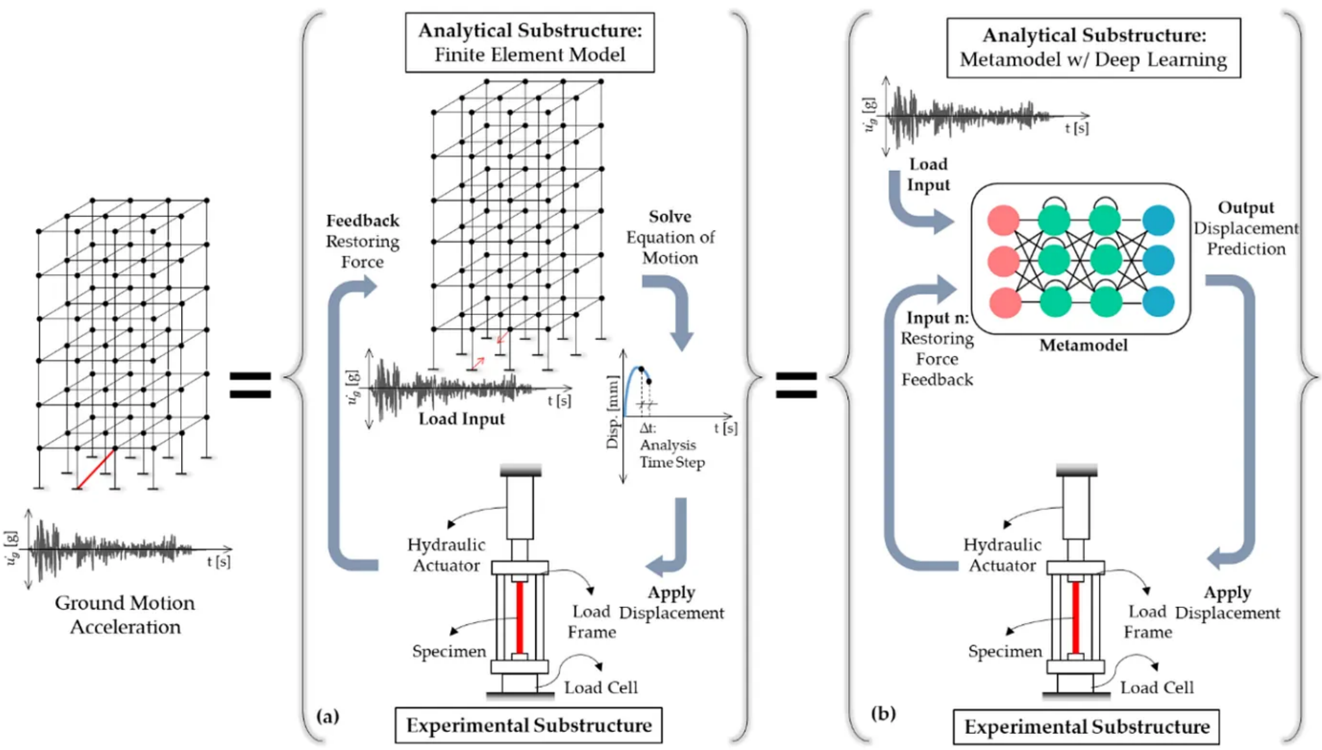


Image Source: https://www.mdpi.com/2504-4990/2/4/26

Introduction

In the seventh volume of their *Traité de psychologie expérimentale*, devoted to the phenomenon of intelligence, Jean Piaget and his co-authors postulated that intelligence manifests itself as the observable outcome of several intellectual activities, activities that belong to the main, broad categories of induction (learning), subsumption (recognition and generalization), deduction (reasoning), and problem solving (Oléron et al., 1963). Accordingly, a definition of artificial intelligence (AI) that complies with the framework could describe AI as the study of machines that are capable of performing any activities that belong to the aforementioned categories. In particular, in the opening essay of that volume, Pierre Oléron pointed out that the intellectual activities rely on the "construction and use of patterns (schemata or models) representing the objects that the subject perceives" (Oléron, 1963) and manipulates. Although in the experimentalist perspective only the stimulus S presented to the subject and the corresponding response R can undergo an empirical investigation, it is fundamental to realize that a number of specialized schemata mediate between S and R, actualizing "the connection between a class of stimuli and a class of responses" (Oléron, 1963). The resemblance of the latter notion to the very process underlying automatic pattern classification (Duda & Hart, 1973) is striking.

Oléron pinpointed a second, fundamental characteristic of intellectual activities, namely their being accomplished through long (or, deep) circuits. This conception transcends the notion of a natural stimulus-reflex reaction pair as observed in all organisms, a notion which (alongside that of conditioned stimulus — conditioned response) is at the basis of classic associationism (Boring, 1950). In the case of the plain stimulus-reflex association, the reflex reaction "follows immediately the presentation of the stimulus according to an organization of inter-connections that is instantly mobilized" (Oléron, 1963). Figure 1 shows a shallow neural network realization of such a basic *stimulus-reflex association*, where the interconnections may be modified (i.e., learned) according to the experience in order to account for new, conditioned input-output associations. To the contrary, long circuits are required in order to realize the détour typical of the intellectual activities. These long circuits connect the natural perception of the stimulus to higher-level schemata which, in turn, are connected to higher levels of abstraction (in terms of other schemata) until a response is eventually formed. Figure 2 shows a deep neural network realizing long circuits. Individual layers (populations of neurons) in the network form patterns of internal representations of the original input stimulus, according to a bottom-up hierarchy of higher-levels of abstraction. The corresponding response is yielded by the output layer. These schemata are learned and refined through experience. Noticeably, when applied to pattern recognition tasks, the schemata represented by the patterns of activation of the internal layers of the deep neural network do literally result in the aforementioned "connection between a class of stimuli and a class of responses," to put it in Oléron's words.

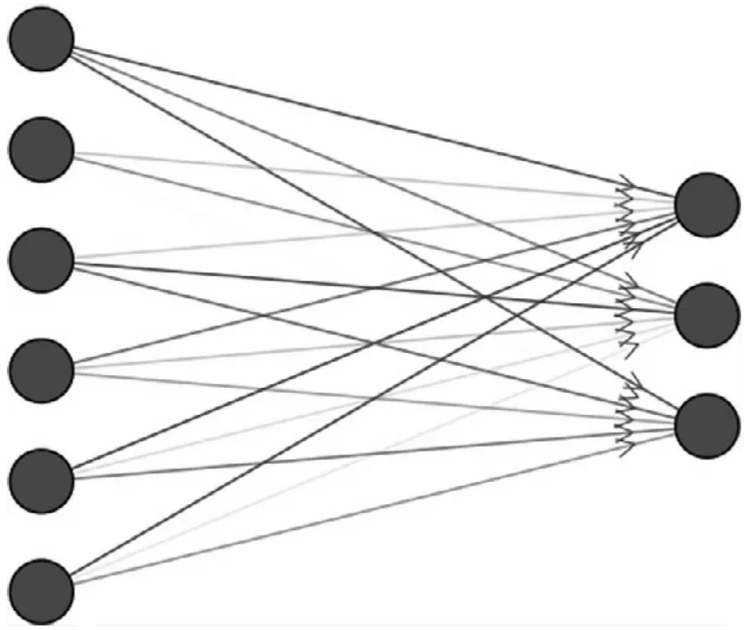


Figure 1: Shallow network realizing a simple stimulus-reflex reaction mechanism (figure generated via NN-SVG (LeNail, 2019)).

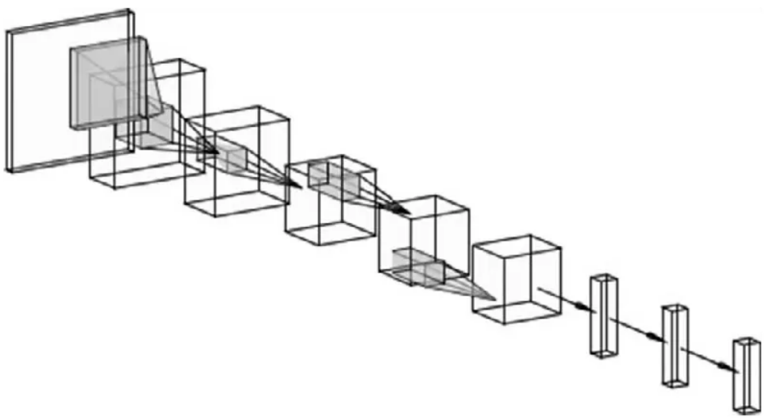


Figure 2: Deep network realizing long circuits among stimulus, schemata, and response (figure generated via NN-SVG (LeNail, 2019)).

. . .

Recent relevant trends in the research on deep learning include graph convolutional networks (Kipf & Welling, 2017), deep networks with attention mechanisms (Cho et al., 2015), generative adversarial networks (Goodfellow et al., 2014b), and deep reinforcement learning (Arulkumaran et al., 2017), among many others. All these advances have been made possible by the impressive, relentless developments in hardware, software libraries, and datasets that have become of everyday use for researchers active in the field. Advances in hardware technology provided deep learning algorithms with a faster and highly parallel processing, thanks to many-core processors, high bandwidth memory, and accelerators suitable to the learning and induction tasks. The most popular form of accelerator is based on the graphics processing unit (GPU), originally devised for fast image manipulation but equipped with processing capabilities that match the computations required in deep neural networks (Steinkrau et al., 2005). Due to the impressive growth in the use of GPUs for deep learning, manufacturers have begun to incorporate neural network-specific instruction sets, or specific tensor cores in their GPUs. Software layers realizing the deep neural network functionalities on GPUs have been developed, as well, and they have become extremely popular among practitioners. Major instances are the libraries TensorFlow (Abadi et al., 2016) and PyTorch (Paszke et al., 2019), among many others. Recently, other forms of accelerators have been proposed (Shawahna et al., 2019), namely field-programmable gate arrays (FPGA) and application-specific integrated circuits (ASIC). Although both FPGAs and ASICs are promising for realizing neural networks, due to their speed and extreme flexibility, they still lack enough momentum to overtake GPUs because of the lack of software layers that can compete with those available for the GPUs. Finally, another factor that contributed to faster development of deep learning lies in the unprecedented effort put by the community into assembling large-scale, real-life datasets whose complexity is challenging enough for constituting sound benchmarks for the new algorithms, and whose size is large enough to allow for learning befitting values for the considerable number of parameters characterizing deep networks without overfitting the training data. OpenML (Vanschoren et al., 2013) and PMLB (Olson et al., 2017) are popular instances of large, public, and curated repositories of benchmark datasets, including software tools for accessing the data in a standardized format.

. . .

Deep Architectures and Representational Issue

Linear and linear-threshold machines construct a map for the input to the output, without any internal representation, thus characterizing the inferential process only by the coefficients of a separating hyperplane in the input space. Inspiration from neuroscience early led to consider feedforward neural architectures which enrich the computation by nonlinear hidden neurons. Interestingly, stacking layers of linear neurons does not increase the computational power of the neural network, since linear layers collapse to a single one. This is clearly the consequence of interpreting the composition of linear functions by the isomorphic matrix product. On the opposite, as we abandon neuron linearity, more sophisticated internal representations of the input arise that are typically referred to as the pattern features. As it will be claimed with more details in the following, once the hidden neurons are organized in layers, a higher degree of abstraction is gained. Interestingly, most interesting human cognitive skills seem to emerge thanks to a sort of natural compositionality, that is in fact at the basis of deep architectures.

When regarding the hidden neurons as units that support appropriate features, one early becomes curious of understanding the secrets behind the pattern of connections generated by learning processes. In the case of fully connected units, one typically expects neurons to construct a very large class of features. Basically, in this case, there are no architectural constraints that, on the opposite, might contribute to gaining invariant properties. Let us consider the classic example of handwritten character recognition task. As for any object recognition task, one very much would like to see neurons developing features that are invariant under scale and roto-translation. In the case of full connections, neurons are developed under the tacit assumption that they are all different from each other, so as they do not support invariant features. An interesting case of invariance arises as the units share the same weights, which also yields a sort of fault-tolerance. Grouping neurons depending on the values of their common weights is a way of forcing the development of features that are translation-invariant. The unit replication, however, becomes more interesting whenever we abandon full connectivity. This is of great importance in vision, where invariance is acquired at different levels. The intrinsic hierarchical nature of deep nets leads to develop neurons acting on small portions of the retina, that are called receptive fields. As we share the weights of neurons operating on receptive fields, we promote the development of translational invariant features, that also gain a hierarchical structure where neurons represent features at different levels of abstraction.

. . .

Internal Representation in Feedforward Networks

The pattern of interconnections in feedforward neural nets (FNN) is defined by a Directed Acyclic Graph (DAG), so as the partial ordering property behind DAGs is the counterpart of the forward data flow mechanism in forward propagation of FNN.

A very interesting special case of the feedforward structure is that of multilayered networks, where the units are partitioned into ordered layers with no internal ordering.

The layered structure dramatically simplifies the data flow propagation of the input. When referring to Figure 3 we can see that the weights associated with a layer can compactly be represented by a corresponding matrix, so as the output turns out to be

$$y = \sigma(\mathrm{W}_3\sigma(\mathrm{W}_2\sigma(\mathrm{W}_1\mathrm{x}))).$$

In general we have

$$\begin{aligned} &\mathrm{x}_0 = \mathbf{u} \\ &\forall l = 1, \ldots, : \mathrm{x}_l = \sigma(\mathrm{W}_{l-1}\mathrm{x}_{l-1}). \end{aligned}$$

Here, the initialization $x_0 = u$ fires the forward propagation step. Of course, the role of σ(·) is crucial in the neural network behavior. The mentioned collapsing to a single layer in case of linearity can be seen. In that case we have σ(·) id(·) and, therefore,

$$y = \prod_{l=1}^{L} \mathrm{W}_\ell \cdot \mathrm{x} = \mathrm{W}\mathrm{x},$$

$$\mathrm{W} := \prod_{l=1}^{L} \mathrm{W}_\ell.$$

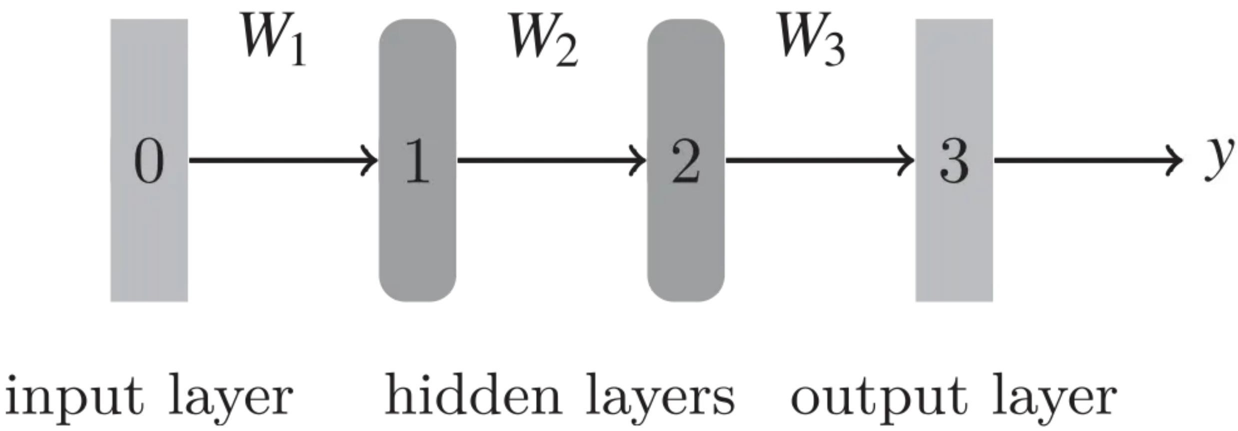


Figure 3: Layered structure with two hidden layers. There is no ordering relationship inside the layers.

We can see that, in general, there is no matrix W3 such that

$$\sigma(\mathbf{W}_2(\sigma(\mathbf{W}_1(\mathrm{x})) = \sigma(\mathbf{W}_3\mathrm{x}),$$

which corresponds with the additional computational power that is gained by nonlinear hidden neurons. The neurons that are modeled by

$$y = g(\mathrm{w}, b, \mathrm{x}) = \sigma(\mathrm{w}'\mathrm{x} + b),$$

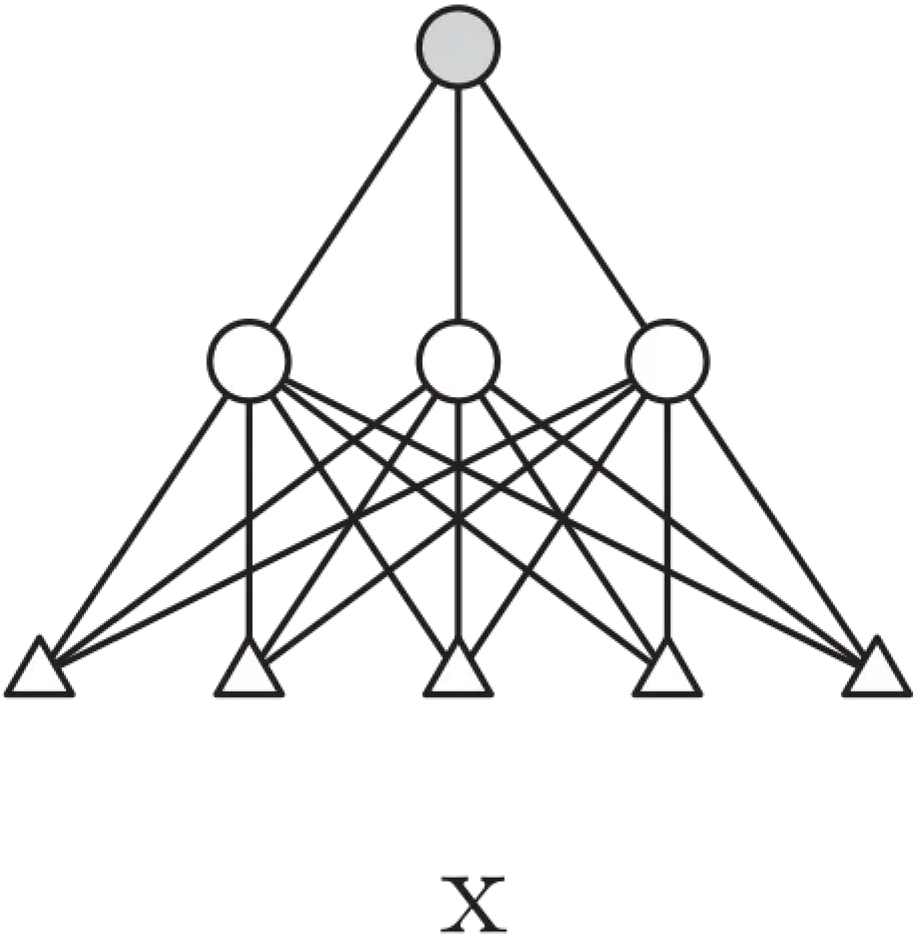
are referred to as ridge neurons. Another classic computational scheme is based on

$$y = g(\mathrm{w}, b, \mathrm{x}) = k\left(\frac{\|\mathrm{x} - \mathrm{w}\|}{b}\right)$$

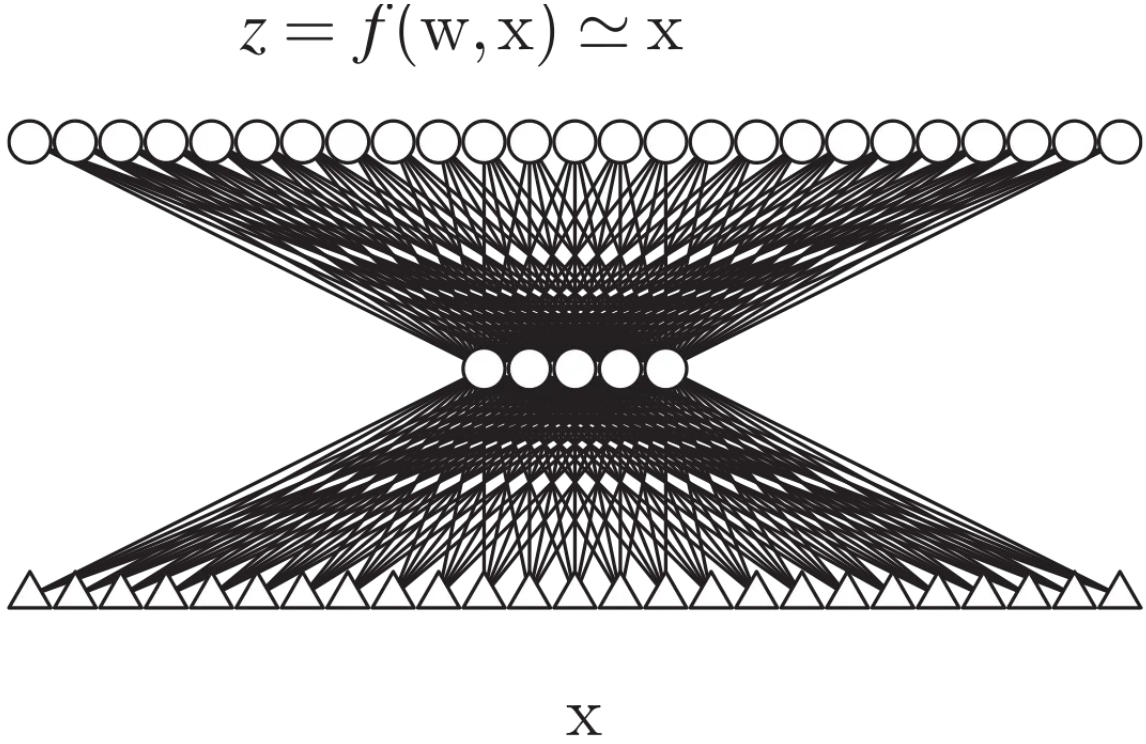
which are called *radial basis function* neurons. Here, *k* is a single-dimensional radial basis function (e.g., a Gaussian function).

The discussion carried out so far has been mostly dominated by the idea of learning agents which interact with the environment according to the supervised-based learning protocol, which is based on imposing that the output z of the neural network gets as close as possible to the target y, that is

$$z = f(\mathrm{w}, \mathrm{x}) \simeq y$$



Most of the cognitive processes that we currently investigate in humans, however, do not rely on such a supervision which provides the target at any stimulus.



A common cognitive skill which is observed in children, and also in other animals, is their ability to learn repeating an input stimulus — think of sound repeating. This suggests the construction of neural networks where the target becomes the input itself so that the network is expected to minimize *e(x(f(w,x))* that is $z = f(w, x) = x$. The encoding architecture extends matrix factorization in linear algebra. In that case, we are given a matrix T and we want to discover factors $W1$,$W2$, so as $T = W2W1$. The process of encoding consists of mapping $x \in \mathbb{R}^d$ to a lower dimension $\bar{y}$, which is the number of hidden units. One would like the network to return $z = f(w, x) \approx x$, so as the output of the hidden neurons can be regarded as a code of the input. Basically, the hidden layer contains an internal representation of the input stimulus in a compressed form.

. . .

Internal Representations of Real-Valued Functions

Real-valued functions share a few analogies with Boolean functions. There are also remarkable differences which are intimately connected with their fundamentally different mathematical structure. Early studies by Lippman and Gold (1987), who assumed to deal with hard-limiting LTU, provided interesting insights on the internal representation of neural networks for classification tasks. In Figure 4, a neural network with two inputs is expected to classify the patterns of a nonconnected domain composed of two convex sets. At the first hidden layers, neurons in 3,4,5 and 6,7,8 can develop connections such that they can represent the two convex sets denoted by 9 and 10, respectively. These convex sets are detected by the corresponding neurons in the second hidden layer. At the output layer, unit 11 can act as a logical disjunction, thus conferring the overall net the task of recognizing any point in the union of the convex sets denoted by 9 and 10. Clearly, the construction shown for non-connected convex sets can be used to realize any concave set.

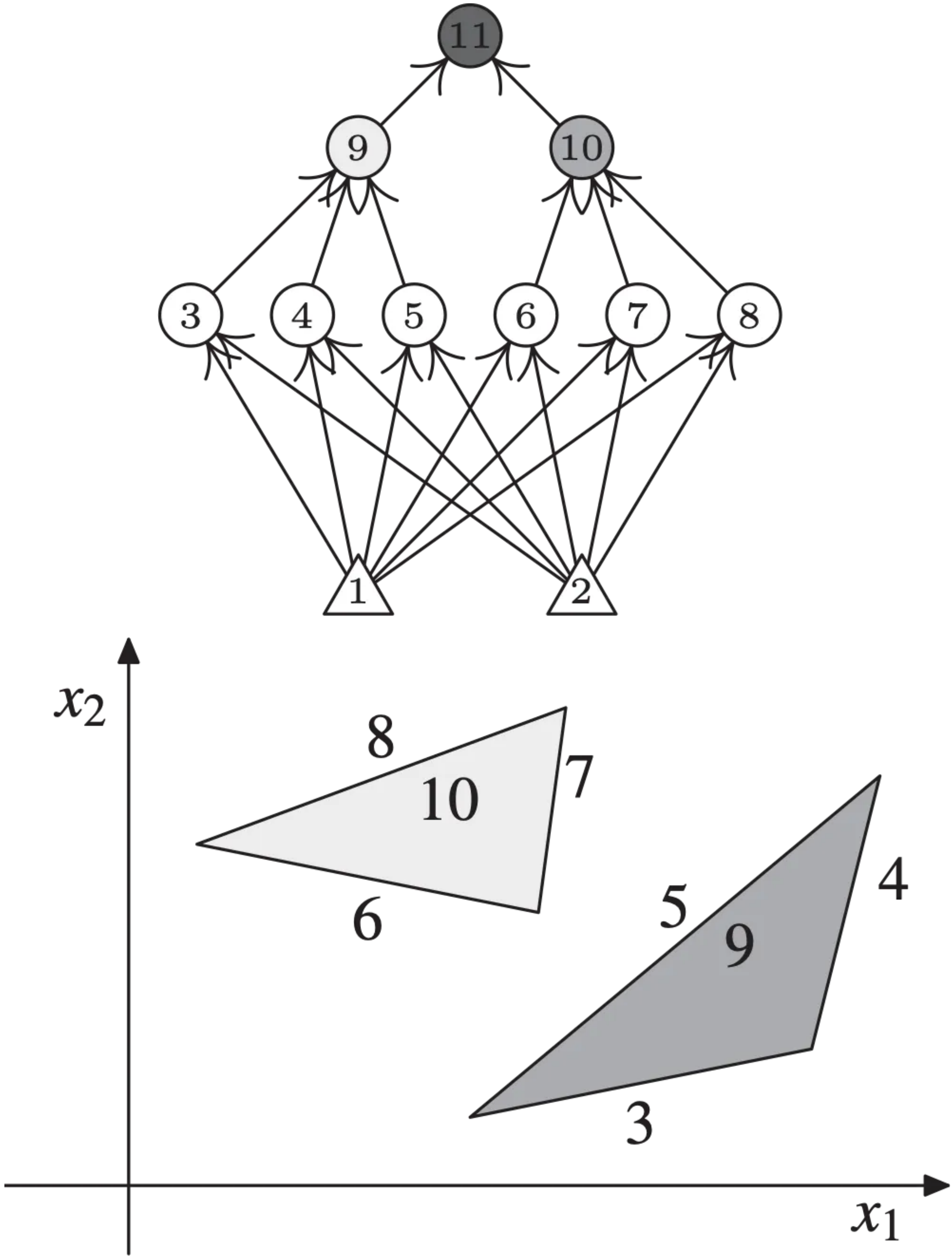


Figure 4: Classification in $\mathbb{R}^2$ using a neural network with hard-limiting units. The non-connected domain $\mathbb{X} = \mathcal{X}_1 \cup \mathcal{X}_2$ is detected by a depth-3 neural network, where at the second hidden layer the convex domains $\mathcal{X}_1$ and $\mathcal{X}_2$ are isolated. Then the or of the output unit represents the characteristic function of $\mathcal{X} = X_1 \cup X_2$

. . .

Convolutional Nets

The discussion on representational issues in the previous section has provided evidence on the importance of abandoning shallow architectures in favor of deep neural nets. The universal computational capabilities that come with the canonical one-hidden-layer architecture turn out to be a mixed blessing. The power of generality is gained by paying the explosive growth of the number of hidden units. On the opposite for deep nets, it has been shown that the number of equivalent configurations drops dramatically in favor of hierarchical architectures that turn out to be more adequate to naturally express most interesting cognitive tasks. Basically, the interest in cognition is not uniformly focused on any possible tasks, but on those which can be experimented in nature. Interestingly, the need to gain abstract concepts to optimize the relationship of intelligent agents with the environment has led to the development of highly structured representations whose interpretation can better be achieved by deep nets. A recurrent important property that is discovered in perceptual tasks is that of invariance. The underlying idea is that different stimuli correspond with the same concepts. An object represented in the retina is the same regardless of its translation, rotation, and scale modification. On the other hand, the supervised learning protocol taking place in shallow networks promotes solutions where the discovery of feature invariance is mostly missed, in favor of the development of multiple representations of the same feature by different neurons. Neural networks which can incorporate invariant features contribute to the development of models that are more suited for the underlying cognitive task.

The most remarkable example of architectures which exhibit built-in (translational) invariance is that of convolutional neural networks. They have had a special role in the revival of neural networks and the advent of deep networks as a technique which revolutionize scientific domains and application fields beyond the domain of computer vision for which it was originally conceived. Historically, the family of convolutional neural networks is based on results from the seminal works of D. H. Hubel and T. Wiesel from the mid-fifties to the late seventies on mammalian visual cortex (Hubel & Wiesel, 1959, 1962, 1977). They described the structure of the visual cortex organized in hierarchical layers of simple cells and complex cells, building complex representations of the visual information from first simple cell responses to specific oriented edges and contrast areas then aggregated, combined, in complex cells. Such simple cells (from the visual cortex) are activated by Gabor-like shape receptive fields (see Figure 5, below).

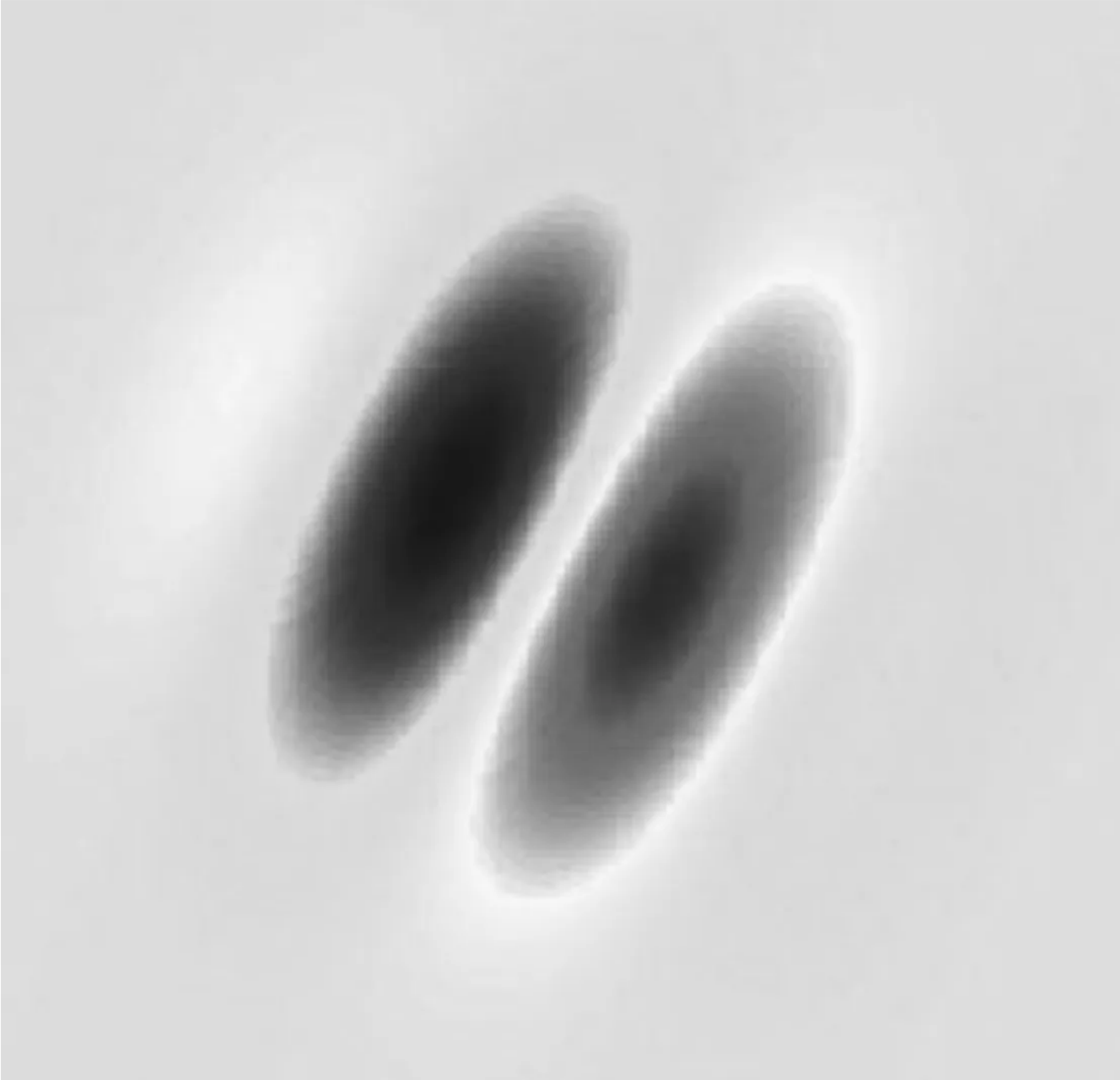


Figure 5: From Wikipedia, "Gabor filter-type receptive field typical for a simple cell.

Based on these results, the first attempt to build a neural network mimicking these mechanisms dates back to studies by K. Fukushima with his cognitron (Fukushima, 1975), and later his neocognitron (Fukushima, 1980). In the latter, in order to make his network invariant to receptive field shifts, and thus invariant to a translation of stimuli, he proposed a very important feature at the core of CNNs: a simple cell is a grid of neurons which all share the same set of weights, but with their "receptive fields" processing the input at different positions (as illustrated in Figure 6). To better understand this mechanism, one simple cell of layer U SL and one complex cell of layer UCL are extracted in Figure 9.10. If this first simple cell in U SL is sensitive to the receptive field (given in Figure 5), then the activation of the resulting neuron in the corresponding complex cell in U CL will be maximal. The weights on the connections are given by the values in the receptive field (each pixel of Figure 5 actually defines a weight). Thus, the input values in the considered region will be multiplied by the aligned corresponding weights of the receptive field. Again, as explained in Fukushima (1980, 2019), since all the neurons of this simple cell share the same weights, the same oblique edge stimulus in U but shifted will activate a neuron with the same magnitude at another location of the complex cell grid of neurons. And if there are several similar stimuli, they will activate all the corresponding neurons in the complex cell similarly. This principle provides shift invariance (or translation invariance) to the activation of a given receptive field. If one looks at the receptive field as a filter, the spatial filtering of the input operated by a simple cell is thus shift invariant. This is precisely, in signal processing domain, the definition of the convolution operation of an input (signal) by a kernel filter (i.e. the receptive field). Such a layer will thus be called a *convolutional layer*.

Figure 6: 0ne simple cell of layer USL and its corresponding complex cell of layer $\mathcal{U}$CL are displayed (the other simple and complex cells at the same levels, or in the same layers, are aligned below as it can be partially seen on the figure).

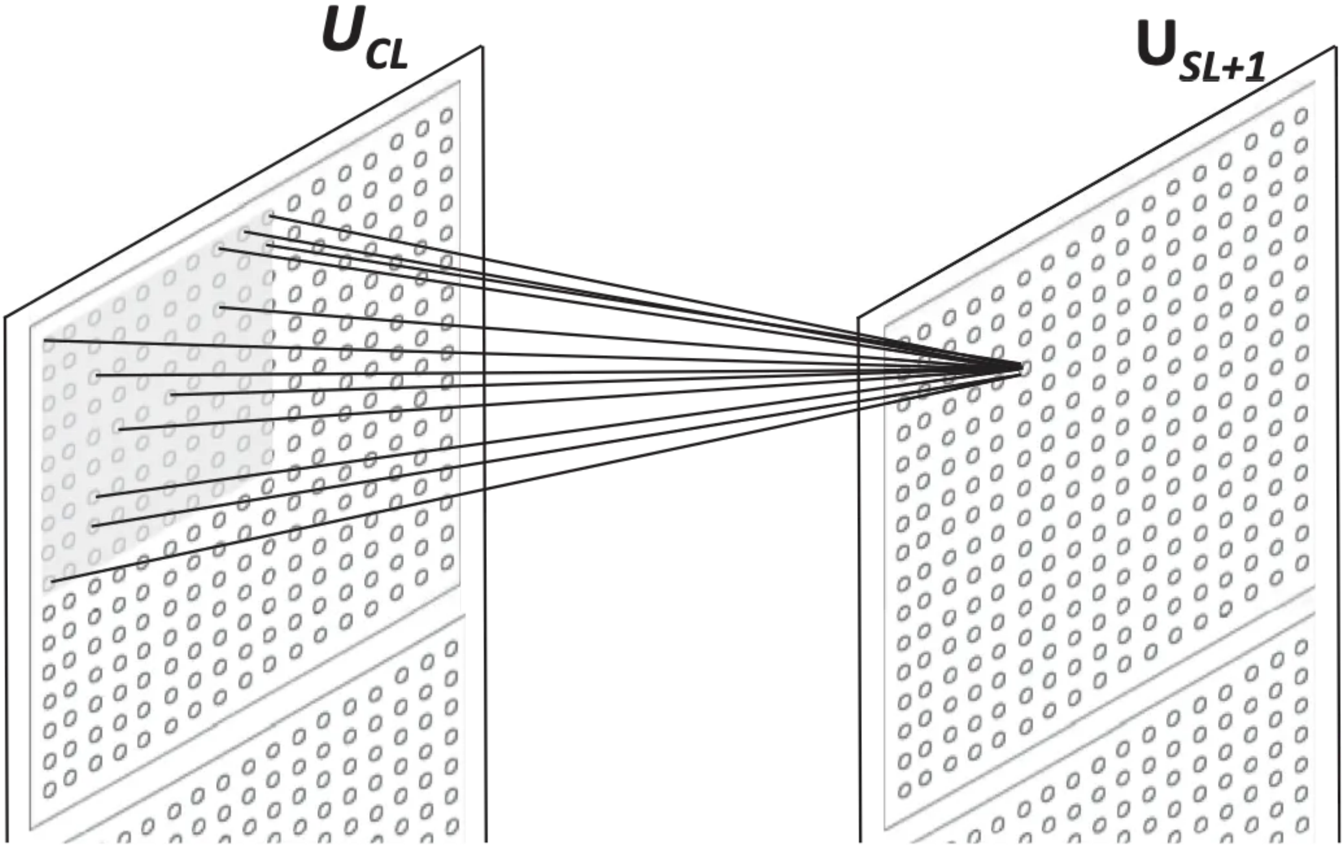


Figure 7: The neuron output in the rectangle area is summarized into one neuron.

The *BackPropagation (BP)* algorithm, did not exist yet at the time and K. Fukushima focused on self-organization map algorithms to optimize the weights of the neocognitron network. This optimization process has the advantage of being unsupervised. However, until now, optimizing deep neural networks using supervised BP algorithm outperforms other strategies.

As aforementioned, Y. LeCun participated in the general effort of the community to conceive BP algorithm and to efficiently train neural networks, in particular LeNet-5, the first CNN with BP and some other adaptations (LeCun et al., 1989). This first CNN model outperformed all other methods for about two decades on the task of digit recognition for the MNIST dataset. This first CNN architecture took several processing steps from the neocognitron as can be seen in Figure 8.

Figure 8: Typical CNN architecture. Reproduced from Wikipedia.

As in the original neocognitron, convolution layers and pooling layers (corresponding to subsampling on Figure 9.12) alternate in CNNs. The subsampling step in most CNN architectures is a max pooling step (preserving only the max value in the pooled area) which provides in addition sparsity on the resulting feature map (see Figure 9).

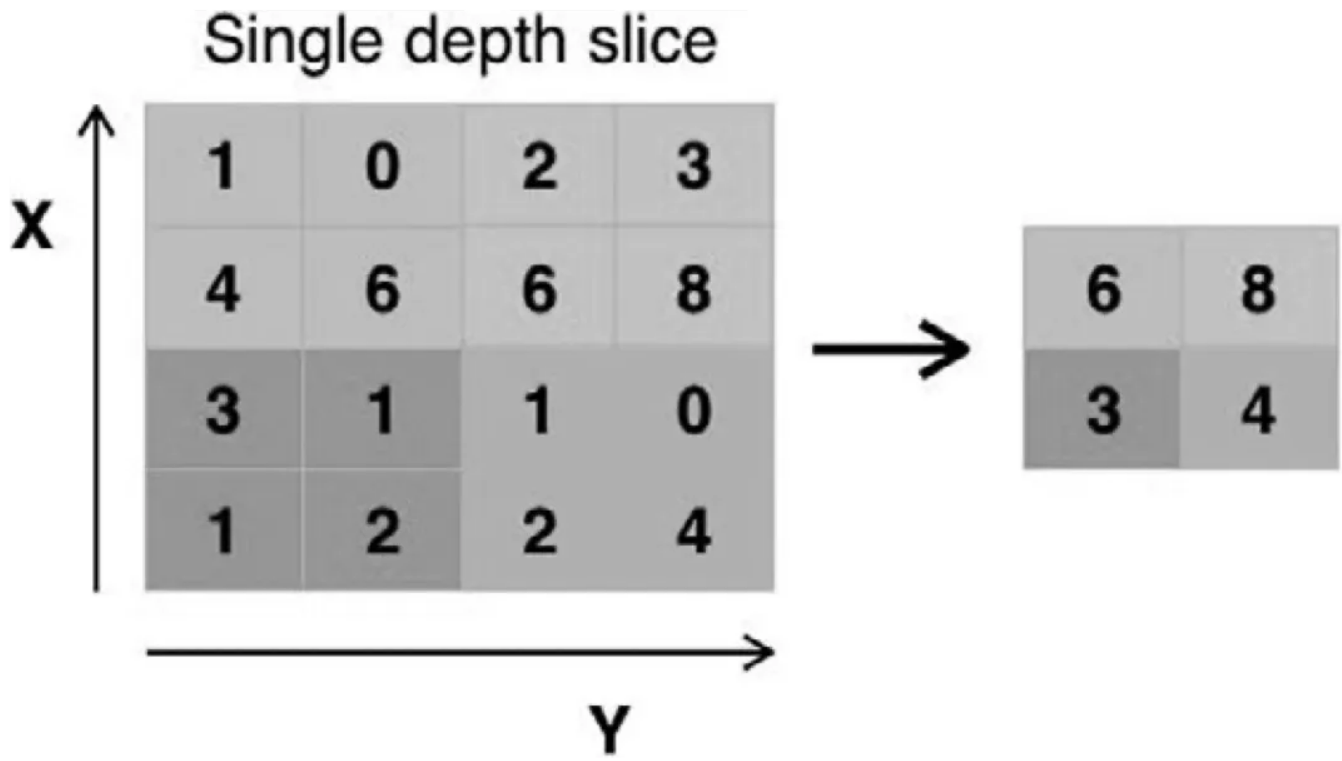


Figure 9: Max pooling with a 2 × 2 filter corresponds to keep only the max value in the 2 × 2 area. In this example, the stride is equal to 2; thus between two consecutive 2 × 2 areas the horizontal and vertical displacements are equal to 2.

A final remarkable feature of the CNNs lies in closing the loop with the origins of the methods: an artificial neuron is a simplistic representation of related biological models. From D. Hubel and T. Wiesel, to K. Fukushima and his neocognitron, to finally Y. LeCun and his LeNet, each new model has increased the distance with biological reality: biological neurons are discrete computational units while artificial ones are continuous computational units; BP or a mechanism mimicking BP has not (yet) been identified in biology; and the main difference lies in the resources required by CNNs to be efficient, such as size of the training set, or amount of energy for training.

. . .

DNNs with Adaptive Activation Functions

Until a few decades ago, neuroscientists agreed on the fact that the neuroplasticity of the human brain, responsible for higher cognitive phenomena like memory and learning, was to be found at the network level, in the pathways of interconnections among neurons and, above all, in the plasticity of the synapses (Fuchs & Flügge, 2014). Phenomena like Hebbian learning (Hebb, 1949) affect the synapses (either excitatory or inhibitory) by strengthening or weakening them, depending on the history of activation of the presynaptic and postsynaptic neurons. Accordingly, in artificial neural networks the focus has long been on learning the "synaptic" connection weights w vu . Starting from the 1980s, several developments in neuroplasticity studies have brought to light phenomena of nonsynaptic plasticity, including morphological and functional modifications of the neuronal cells that occur in parallel with changes of the synapses (Mozzachiodi & Byrne, 2010). Such modifications are mostly related to the intrinsic capability of a neuron to adjust its own excitability (*homeostatic plasticity*), that is the function it realizes, in response to (and, in compensation for) the activity of neural pathways embracing that neuron. In particular, homeostatic scaling consists in a modification of the action potential of the neuron such that "the neuron increases the strength of all excitatory connections in response to a prolonged drop in firing rates, and vice versa" (Turrigiano & Nelson, 2000), substantially "scaling synaptic transmission in a multiplicative manner by a negative feedback mechanism (. . .) while preserving relative synaptic weight encoded in individual synapses and thus memory information" (Siddoway et al., 2014).

At the same time, learning algorithms for artificial neural networks that comprised the adaptation of the activation functions realized by the artificial neurons began to flourish, leading to improved performances of the resulting machines. The vast majority of these algorithms revolved around the idea that the activation functions could be expressed in a parametric form, and that the specific value of the corresponding parameters could be learned from the data. Early attempts centered on the parameters b and σ of logistic sigmoids having form (as shown in Figure 10), where the bias b determines the location of the sigmoid and σ affects its slope. In recent years, researchers have been investigating several parameterized variants of the rectifier linear unit (ReLU) activation function for DNNs in the form $f(a) = \lambda g(a)$, where $g(\cdot)$ is a base transformation (e.g., a hinge function, that is $g(a)= max(0, a)$ and λ is a real-valued parameter that may be tuned empirically or adapted autonomously as part of the DNN learning process. Prominent examples are the leaky ReLU with adaptive slope λ (that is f(a) = λa if a ⩽ 0) (He et al., 2015) and its stochastic variant (Xu et al., 2015), the exponential linear unit (ELU) (that is a rectifier with *f(a) = λa* if *a ≤ 0*) (Clevert et al., 2016), as well as the scaled ELU (SELU) (Klambauer et al., 2017) where the ELU is multiplied by λ regardless of a being positive or negative. It is seen that all these adaptive activation functions are special cases of the general algorithm for learning λ (originally presented by Trentin (1998)) that is covered in the present section.

$$f(a) = 1/(1 + \exp(-(a - b)/\sigma))$$

Figure 10

. . .

Conclusions

This article has covered topics of deep learning in artificial neural networks, putting an emphasis on the particular experimentalist perspective that underlies implicitly this field of research. Deep learning was positioned in the proper historical perspective, mentioning first nonneural machine learning paradigms that have long been established as suitable models of hierarchies of higher levels of representation of the input stimuli, and then pointing out the milestones of the half-century-long path that led scientists to develop deeper and deeper neural network architectures. Major paradigms were mentioned (e.g., stacked autoencoders) or presented in detail (convolutional neural networks). Nowadays, the field has broadened to such an extent that an in-depth survey of the state of the art would have required much more than a single blog (readers are referred to the textbook Deep Learning by Ian Goodfellow, Yoshua Bengio, and Aaron Courville). The present authors preferred to get deeper into some specific, fundamental issues, in particular representational properties of deep architectures and homeostatic neuroplasticity by means of adaptive activation functions. The topic has been inspired by recent developments in neuroscience, and it has been the focus of many studies throughout the last twenty years, resulting in improved DNN learning and generalization capabilities.

For the years to come, the field is expected to develop further, having become the hotspot of research in AI and allied sciences. Scientists worldwide are on their way towards larger and deeper architectures, novel algorithms, all sorts of practical techniques to improve and expedite the learning process, and (above all) a number of significant real-life applications. The developments in DNN research have been triggered by (and will continue to proceed jointly with) the increase in computational power, an increase due to the advancements in hardware technologies, in particular the advent and progress of GPUs (graphics processing units). The alliance between GPUs and DNNs is here to stay, at least for the next decade.

. . .

References (in order of citation):

Oléron, P., Piaget, J., Inhelder, B., & Gréco, P. (1963). Traité de psychologie expérimentale VII. L'Intelligence. Paris: Presses Universitaires de France.

Duda, R. O., & Hart, P. E. (1973). Pattern Classification and Scene Analysis. New York, NY: Wiley.

Boring, E. (1950). A History of Experimental Psychology. New York, NY: Appleton-Century-Crofts.

Oléron, P. (1963). Les activités intellectuelles. In P. Oléron, J. Piaget, B. Inhelder, & P. Gréco, (Eds.), Traité de psychologie expérimentale VII. L'Intelligence (pp. 1–70). Paris: Presses Universitaires de France.

LeNail, A. (2019). NN-SVG: publication-ready neural network architecture schematics. The Journal of Open Source Software, 4(33), 747.

Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In Proceedings of the 5th International Conference on Learning Representations. OpenReview.net.

Cho, K., Courville, A., & Bengio, Y. (2015). Describing multimedia content using attention-based encoder-decoder networks. IEEE Transactions on Multimedia, 17(11), 1875–1886.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., et al. (2014b). Generative adversarial nets. In Z. Ghahramani et al., (Eds.), Advances in Neural Information Processing Systems, 27, 2672–2680.

Arulkumaran, K., Deisenroth, M. P., Brundage, M., & Bharath, A. A. (2017). Deep reinforcement learning: a brief survey. IEEE Signal Processing Magazine, 34(6), 26–38.

Steinkrau, D., Simard, P. Y., & Buck, I. (2005). Using GPUs for machine learning algorithms. In Proceedings of the 8th International Conference on Document Analysis and Recognition (pp. 1115–1119). IEEE Computer Society.

Abadi, M., Barham, P., Chen, J., et al. (2016). Tensorflow: a system for large-scale machine learning. In K. Keeton, & T. Roscoe, (Eds.), In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (pp. 265–283). USENIX Association.

Paszke, A., Gross, S., Massa, F., et al. (2019). PyTorch: an imperative style, high-performance deep learning library. In H. Wallach et al. (Eds.), Advances in Neural Information Processing Systems 32, (pp. 8024–8035). New York, NY: Curran Associates.

Shawahna, A., Sait, S. M., & El-Maleh, A. (2019). FPGA-based accelerators of deep learning networks for learning and classification: A review. IEEE Access, 7, 7823–7859.

Vanschoren, J., van Rijn, J. N., Bischl, B., & Torgo, L. (2013). OpenML: networked science in machine learning. SIGKDD Explorations, 15(2), 49–60.

Lippmann, R. P., & Gold, B. (1987). Neural classifiers useful for speech recognition. In IEEE Proceedings of the First International Conference on Neural Networks, vol. IV (pp. 417–422). San Diego, CA.

Hubel, D. H., & Wiesel, T. N. (1959). Receptive fields of single neurones in the cat's striate cortex. The Journal of Physiology, 148(3), 574.

Hubel, D. H., & Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. The Journal of Physiology, 160(1), 106.

Hubel, D. H., & Wiesel, T. N. (1977). Ferrier lecture-functional architecture of macaque monkey visual cortex. Proceedings of the Royal Society of London. Series B. Biological Sciences, 198(1130), 1–59.

Fukushima, K. (1975). Cognitron: a self-organizing multilayered neural network. Biological Cybernetics, 20(3–4), 121–136.

Fukushima, K. (1980). Neocognitron: a self organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics, 36(4), 193–202.

Fukushima, K. (2019). Recent advances in the deep CNN neocognitron. Nonlinear Theory and Its Applications, IEICE, 10(4), 304–321.

LeCun, Y., Boser, B., Denker, J. S., et al. (1989). Backpropagation applied to handwritten zip code recognition. Neural Computation, 1(4), 541–551.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. In Proceedings of the IEEE, pp. 2278–2324.

Fuchs, E., & Flügge, G. (2014). Adult neuroplasticity: more than 40 years of research. Neural Plasticity, 541870, 1–10.

Hebb, D. O. (1949). The Organization of Behavior: A Neuropsychological Theory. New York, NY: Wiley.

Mozzachiodi, R., & Byrne, J. (2010). More than synaptic plasticity: role of non-synaptic plasticity in learning and memory. Trends in Neurosciences, 33(1), 17–26.

Turrigiano, G. G., & Nelson, S. B. (2000). Hebb and homeostasis in neuronal plasticity. Current Opinion in Neurobiology, 10(3), 358–364.

Siddoway, B., Hou, H., & Xia, H. (2014). Molecular mechanisms of homeostatic synaptic downscaling. Neuropharmacology, 78, 38–44.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In Proceedings of the 2015 IEEE International Conference on Computer Vision, (pp. 1026–1034). IEEE Computer Society, USA.

Clevert, D., Unterthiner, T., & Hochreiter, S. (2016). Fast and accurate deep network learning by exponential linear units (elus). In Y. Bengio, & Y. LeCun, (Eds.), Proceedings of the 4th International Conference on Learning Representations (ICLR, 2016).

Klambauer, G., Unterthiner, T., Mayr, A., & Hochreiter, S. (2017). Self-normalizing neural networks. In I. Guyon et al., (Eds.), Advances in Neural Information Processing Systems 30 (pp. 971–980).

Trentin, E. (1998). Learning the amplitude of activation functions in layered networks. In M. Marinaro, & R. Tagliaferri (Eds.), Neural Nets — WIRN Vietri 98, vol. 7081 of Lecture Notes in Computer Science, (pp. 138–144). Berlin: Springer.

. . .

Thank you so much for your attention. Comments are highly welcome! Stay tuned for more!

Alireza Dehbozorgi

https://www.linkedin.com/in/alireza-dehbozorgi-80557702a/

Twitter: **@BDehbozorgi83**

Email: **alirezadehbozorgi83@yahoo.com**

Cognitive Science Computation Deep Learning Linguistics Neuroscience

Written by Alireza Dehbozorgi

I'm a linguist and AI researcher interested in mathematical/computational approaches to both human and formal languages. Twitter: @BDehbozorgi83

More from Alireza Dehbozorgi

Alireza Dehbozorgi

Is a 6D Representation of Meaning Possible?

Preamble

4 min read · May 28

Alireza Dehbozorgi

Computational Models of Cognition: Models of Inductive Reasoning

Causal inference is the study of how actions, interventions, or treatments affect outcomes of interest.

36 min read · Jun 4

55 2

Alireza Dehbozorgi

Hopf Algebras and the Mathematical Structures of Morphosyntax

Introduction

4 min read · Jun 3

8 1

Alireza Dehbozorgi

Some Thoughts on Integrating Acoustic Speech Data into Formal Ontologies and Knowledge Management…

Acoustic analysis can be used to integrate speech cues and data into formal ontologies (1) by identifying and extracting acoustic features…

6 min read · May 24

101 2

See all from Alireza Dehbozorgi

Recommended from Medium

Cory Doctorow

How Amazon transformed the EU into a planned economy

Monopsony, junk fees and market power.

· 11 min read · Jun 14

3.1K 38

Enrique Dans in Enrique Dans

Now we know: the US government carries out surveillance on a scale the Stasi could only dream of

Democratic Senator Ron Wyden's request for the declassification of documentation held by the Office of the Director of National…

· 3 min read · Jun 14

1.99K 24

Lists

Natural Language Processing

345 stories · 7 saves

Practical Guides to Machine Learning

10 stories · 23 saves

Now in AI: Handpicked by Better Programming

248 stories · 11 saves

Self-Improvement 101

20 stories · 184 saves

Darius Foroux

5 Stoic Lessons on Happiness I Wish I Knew Sooner

I don’t have a time machine, but by sharing these lessons with you, I hope I can help you to save time

· 6 min read · 5 days ago

2.6K 55

Rosie Hoggmascall in UX Collective

I cancelled my Calm membership. Here’s all the emails they sent me

A lesson in what not to do with CRM strategy and monetisation

· 8 min read · Jun 12

2.9K 44

Attila Vágó

There’s More To The Reddit Meltdown Than Meets The Eye

A cross-functional, objective analysis of the Reddit scandal…

· 10 min read · 2 days ago

483 9

Nick Wignall

4 Mental Habits that Cause Low Self-Esteem

#2: Worry about the future

· 7 min read · 2 days ago

1K 32

See more recommendations