International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 07 | July - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

Optimizing Image Recognition: Advancements through Conditional Deep Learning for Energy Conservation

Sujeet Kumar Nayak

Synergy Institute of Technology, Bhubaneswar

Abstract - Deep learning frameworks utilizing neural networks have emerged as powerful tools for various recognition tasks across contemporary digital platforms. Despite their effectiveness, the high computational demands of these networks call for energy-efficient solutions. It's been noted that not all inputs necessitate the full processing power of the network; many can be accurately recognized with minimal computation. This paper introduces a concept called Adaptive Deep Learning (ADL), which leverages the features of convolutional layers to gauge the complexity of input data and selectively engage subsequent network layers. This is accomplished by integrating a sequential linear neuron network at each convolutional stage, using its output to determine if the classification process can conclude at that juncture. This approach allows the network to tailor its computational load to the input's complexity, without compromising on accuracy.

For datasets such as MNIST, CIFAR10, and Tiny ImageNet, the energy savings realized by employing cutting-edge deep learning structures are significant, with reductions of 1.84x, 2.83x, and 4.02x, respectively. Additionally, this conditional technique is applied to the foundational training of deep learning networks, incorporating direct feedback from extra output neurons positioned at the mid-level convolutional layers. The integrated ADL training method proposed here enhances the rate of gradient convergence, leading to a marked decrease in error rates for MNIST and CIFAR-10 and yielding superior classification performance compared to conventional baseline networks.

Key Words: optics, photonics, light, lasers, templates, journals

1.INTRODUCTION

Traditional deep learning models typically process each input through the entire depth of the network to reach a classification decision. Yet, the complexity of inputs in real-world datasets varies significantly. For instance, distinguishing a person against a solid-colored background is far simpler than identifying them within a crowded scene. To enhance both speed and energy conservation, the computational resources expended by algorithms should align with the input complexity, as suggested by Venkataramani and colleagues in 2015. This study introduces a novel approach termed Adaptive Layer Engagement (ALE) for deep learning networks (DLNs), which employs a tiered system to selectively activate deeper layers based on the input's complexity, thereby achieving quicker and more energy-efficient processing.

This paper also highlights an interesting characteristic of convolutional neural networks (CNNs) within DLNs, which serve as visual processing layers. These CNNs develop a feature hierarchy that evolves from basic patterns (akin to Gabor filters and color blobs as noted by Zeiler et al. in 2010) to more intricate ones with increasing network depth (Yosinski et al., 2014). DLN models, especially those trained for classification tasks, have been repurposed as feature extractors by omitting the final output layer (Razavian et al., 2014; Szegedy et al., 2015; He et al., 2015b). Notably, features derived from a pre-trained DLN, such as OverFeat (Sermanet et al., 2013), have proven effective in computer vision applications like scene recognition and object detection. In this context, the paper leverages the broad-to-detailed feature progression in CNN layers to discern the varying difficulty levels of dataset inputs. The initial CNN layers' outputs are utilized to classify simpler instances without engaging the full network. Conversely, only the more challenging cases, which typically represent a minor portion of the dataset, activate the deeper layers for a more precise classification outcome.

Deep Learning Networks (DLNs), akin to other methods of supervised learning, operate in two distinct modes: training and evaluation. During the training stage, the model learns to demarcate decision boundaries using the provided labels. In the evaluation stage, this trained model is then applied to classify unseen data. The core principle of Adaptive Computational Engagement (ACE) is to construct a sequence of decision-making models at each convolutional layer during training, diverging from the conventional singular complex model approach. When testing, the complexity of the input dictates the number of models or layers engaged for precise classification.

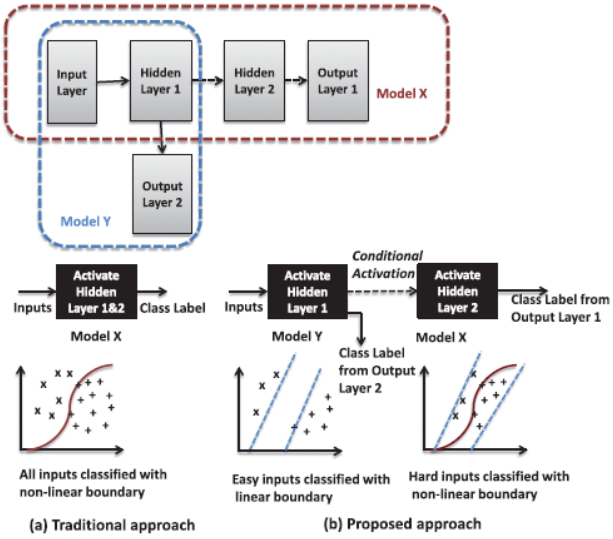


Fig. 1. (a) Traditional approach where both layers are activated and all inputs are classified with a nonlinear
boundary. (b) Proposed approach where easy instances are classified at hidden layer 1 with linear
boundary and hard instances at layer 2 with a non-linear boundary [Venkataramani et al. 2015].

To illustrate, consider a two-layer neural network classifier. The conventional method, depicted in Figure 1(a), processes all inputs through the complex model X, which may necessitate multiple hidden layers for non-linear boundaries, thus increasing computational load. This model activates both hidden layers for high-accuracy classification, leading to unnecessary computation for simpler inputs. Our proposed ACE methodology, shown in Figure

1(b), introduces two decision models, Y and X. Model Y, activated by the first layer alone, classifies straightforward inputs by forming a hyperplane around the non-linear boundary. If the confidence of the output from the first layer falls below a certain threshold δ, the more intricate model X is engaged, activating the second hidden layer for complex inputs. This strategy significantly conserves energy, as not all instances require the elaborate non-linear model X. Furthermore, our experiments reveal that ACE not only conserves computational resources but also enhances accuracy compared to the baseline DLN. It's important to note that ACE refines a pre-existing baseline model by adding output layers, thereby reducing computational demands during testing.

Motivated by the promising results in accuracy enhancement, our research ventured into a novel territory with Integrated Training using Conditional Deep Learning (CDL), as detailed in Section 6. This exploration involved harnessing the additional output layers for training a Deep Learning Network (DLN) from the ground up. DLNs have historically encountered obstacles during training, notably the issue of diminishing gradients, as identified by Glorot and Bengio in 2010. CDL capitalizes on the strengths of Convolutional Neural Network (CNN) features and these extra output layers to streamline the DLN's testing phase. In contrast, Integrated CDL (ICDL) training utilizes the guidance provided by these layers to enhance DLN accuracy and alleviate training challenges.

The field has seen various innovative training strategies, such as data augmentation, dropout, maxout, and layerwise pre-training, which have significantly boosted DLN performance on complex tasks. Building upon the work of Szegedy et al. in 2015, who incorporated two additional output layers in a 22-layer DLN (GoogleLeNet) for better regularization and gradient propagation, we introduce Integrated CDL training. This method, an evolution of our prior research, aims to refine DLN learning processes further. Our experiments indicate that Integrated CDL markedly betters gradient convergence and minimizes error rates.

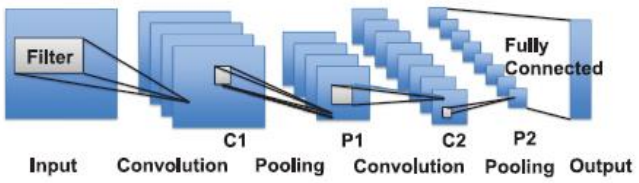


Fig. 2. A standard architecture of a Deep-Learning Convolutional Network.

Although our training approach is inspired by Szegedy et al., it is distinct in its focus, design, and evaluation techniques. The concept of adding extra output layers was initially proposed in our earlier work. However, that preliminary study on CDL was limited to the MNIST dataset for digit recognition. In the current research, we expand CDL's application to a broader range of datasets and more complex state-of-the-art DLNs, confirming our methodology's efficacy in addressing intricate recognition challenges. We examine the CDL architecture across renowned networks like LeNet, AlexNet, and ResNet, and datasets such as MNIST, CIFAR10, and Tiny ImageNet, utilizing the Torch platform. Our findings demonstrate that our proposed approach not only enhances accuracy but also significantly curtails runtime and computational expenses during testing. Furthermore, we introduce an optimized training regimen based on CDL's observed accuracy gains. This regimen, known as Integrated CDL training, boosts DLN accuracy through additional oversight from the intermediate output layers during the training phase. By backpropagating error gradients from these layers, alongside the final output layer, we achieve a more favorable gradient convergence pattern.

The remainder of the article is structured as follows: Sections 2 and 3 outline the foundational design of a CDL network with an emphasis on energy efficiency. Section 4 details the experimental framework for assessing the CDL network. Section 5 discusses the observed energy savings and corresponding accuracy improvements with CDL. Finally, Section 6 delves into the Integrated CDL training methodology, with a primary focus on enhancing accuracy.

2. Conditional deep learning classification

In this discourse, we delineate the systematic methodology employed to construct the envisioned Conditional Deep Learning Network (CDLN). As previously established, Convolutional Neural Networks (CNNs) are the cornerstone of a deep learning network (DLN), which typically comprises multiple convolution and max pooling layer pairs, as described by LeCun and colleagues in 1998. A basic DLN architecture, illustrated in Figure 2, consists of convolutional layers (C1, C2) succeeded by pooling layers (P1, P2). The convolutional layer applies a set of weight kernels across the preceding layer to generate a series of output maps. These kernels are systematically applied over the entire input field. Subsequently, a max-pooling layer reduces the dimensionality of the convolution layer's activations by selecting the highest activation within a segment of the prior layer's map, thereby instilling translational invariance to minor pixel shifts in input images. The deeper layers, equipped with an increased number of kernels, process the more intricate elements of the image on reduced-dimensional inputs. The terminal fully connected layers amalgamate inputs from all preceding maps to execute the comprehensive classification of the input data. This layered hierarchy has proven effective for image recognition tasks, as noted by LeCun et al. in 2004.

Building on the premise that CNN layers in DLN models, when trained for classification, can serve as feature extractors upon removal of the output layer, we harness the power of convolutional layer features to forge an architecture where simpler instances are classified at earlier stages, thus bypassing the activation of subsequent DLN layers. This approach not only streamlines the classification process but also optimizes computational efficiency by reducing unnecessary processing for less complex inputs.

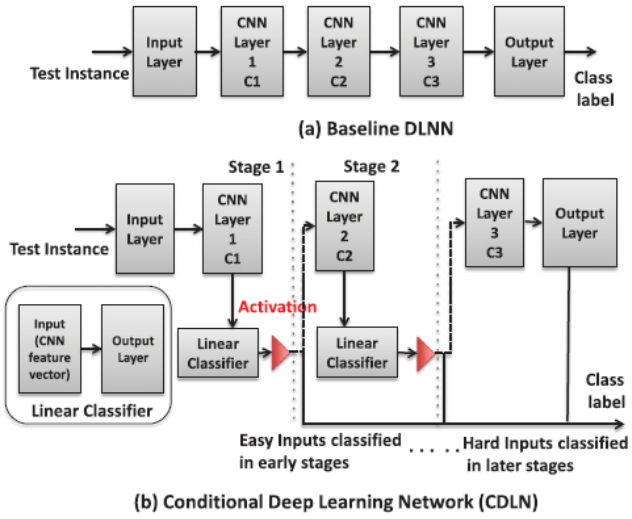


Fig. 3. (a) Baseline deep-learning network. (b) CDLN with linear classifiers added at the convolutional layers whose output is monitored to decide if classification can be terminated at current stage or not.

Figure 3 presents a visual representation of the Conditional Deep Learning Network (CDLN) concept. In Figure 3(a), the foundational deep learning network is depicted, comprising three convolutional layers (C1, C2, C3), which are trained using the conventional

backpropagation technique. To simplify the illustration, pooling layers and filters are omitted.

Figure 3(b) demonstrates the proposed cascaded structure. Here, the outputs from each convolutional layer are directed to a linear classifier. These classifiers have an equivalent number of output neurons as the final output layer in the baseline DLN (as shown in Figure 3(a)). Consequently, the proposed CDLN is composed of multiple stages, each corresponding to a CNN layer, and linked sequentially. Each stage includes a linear classifier that is trained using the features from its respective convolutional layer. The activation of subsequent stages in the CDLN is contingent upon the results from the linear classifiers at each stage.

As the network delves deeper, the decision boundary models at each CDLN stage become increasingly complex and non-linear. Therefore, simpler inputs are classified at the initial stage, while more complex inputs are processed in the later stages for accurate classification. This tiered approach allows for efficient computation by adapting the level of processing to the complexity of the input, thereby optimizing both speed and energy usage.

In simpler terms, think of the process as a series of checkpoints, each with a guard (the activation module) who checks your ID (the input instance). If your ID isn't clear enough or matches too many people (low confidence in class labels), the guard sends you to the next checkpoint for further verification. But if your ID is clear and matches only you (high confidence in one label), you're given the all-clear, and you don't need to go any further. This system ensures that only the instances with clear, strong identification are quickly processed, while the more complex cases get extra scrutiny.

To put it in a more relatable way, imagine you're improving a recipe. The original recipe (the baseline DLN) is good, but you've found a way to make it even better by adding a special ingredient (the linear networks). This ingredient is easy to prepare and mixes well because it's based on the flavors already present in the dish. By incorporating this new ingredient, you end up with a dish (the CDL network) that not only tastes better (higher classification accuracy) but also cooks faster (energy efficiency). Even if the original recipe wasn't perfect, this special ingredient helps to enhance the overall flavor, making your dish stand out in a cooking competition (competitive classification accuracy).

2.1. Efficiency and Accuracy Optimization (*Adding Linear Classifiers at the Convolutional Layers.)*

First, we examine whether it is desirable to add a linear classifier for every convolutional layer of the DLN. Please note that we need to take into account the additional cost of adding an output layer of neurons for each convolutional layer while calculating energy costs [Venkataramani et al. 2015]. Let the computational cost of the CDLN at a particular stage (including the additional cost of linear network at that stage) i be γi per instance. Let the fraction of instances that reach stage i be Ii. Similarly, the fraction of instances that reach stage i 1 is Ii 1. Thus, the stage i classifies only a smaller subset (Ii Ii 1) of the inputs. Stage i should satisfy Equation (1) shown below in order to improve the overall efficiency of the framework,

$$(\gamma_{i+1} - \gamma_i).(I_i - I_{i+1}) > \gamma_i.I_{i+1}.$$

3. Benefits with CDL

In this section, we present the experimental results that establish the potential of CDL. We use MNIST (with LeNet as baseline DLN) as our primary benchmark to evaluate the benefits with CDL with respect to energy and accuracy.

3.1. Energy Improvement

Imagine you're looking at a report card that shows how much students (different digits) have improved in their studies (efficiency) compared to last year (standard DLN models). In this report, represented by Figure 6, we're focusing on a particular group of students (20 classes from Tiny ImageNet using AlexNet-8) for ease of understanding.

The report measures improvement by counting how many questions (operations or computations) each student can solve per test (per input). It turns out that one student, representing the number 1, has shown the most improvement, with scores ranging from **1.50 to 2.32 times better** than last year, averaging at **1.91 times** better. This suggests that the number 1 is a quick learner, easily grasping concepts (easier instances) that don't require much explanation (early layers of classification).

On the other hand, the student representing the number 5 didn't show as much improvement and needed more help (activation of deeper layers) to understand the lessons (accurate classification). This is because the lessons for number 5 were more complex (closer to the non-linear decision boundary).

Overall, the new teaching method (AlexNet_CDL) used for the CIFAR dataset shows that, on average, students are doing **2.85 times better** than with the old method (AlexNet-8). This means the new method is not only helping students learn faster but also understand the material better, leading to a significant boost in their performance.

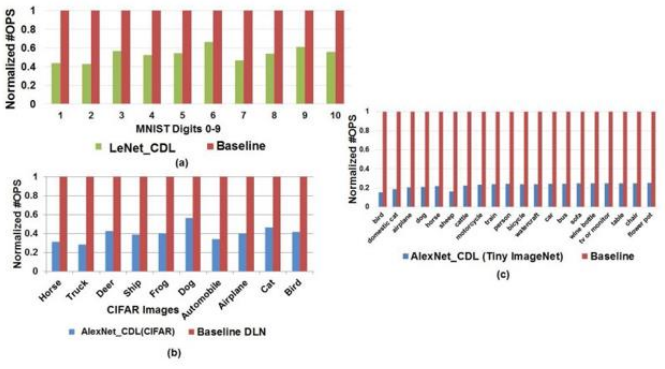
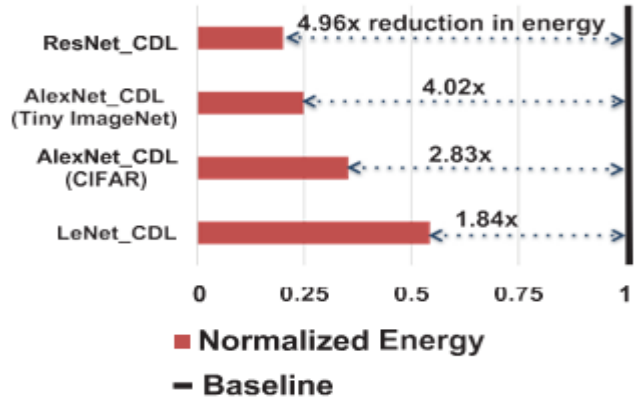


Fig: 3(a) and 3(b)



Network	Time (ms)	Gain	Fraction (%) of instances classified at the additional layers (O1, O2,...) and final output layer (FC)
LeNet-5 (MNIST)	3.31	–	–
LeNet.CDL	0.78	4.24×	82%, 9.2%, 4.8%
AlexNet-8(CIFAR10)	9.42	–	–
AlexNet.CDL(CIFAR)	5.13	1.83×	69.6%, 18.8%, 11.6%
AlexNet-8(Tiny ImageNet)	23.87	–	–
AlexNet.CDL(Tiny ImageNet)	17.31	1.37×	48.4%, 14.4%, 8.3%, 6.1%, 12.8%
ResNet-50 (Tiny ImageNet)	285.20	–	–
ResNet.CDL	166.8	1.71×	31.4%, 10.1%, 9.2%, 7.1%, 6.5%, 5.9%, 5.2%, 6.3%, 18.3%

Table I. Performance Results for Different CDLN Structures

Network	Baseline	CDLN
LeNet (MNIST)	97.55%	98.92%
AlexNet-8 (CIFAR)	78.38%	79.19%
AlexNet-8 (Tiny ImageNet)	65.24%	66.14%
ResNet-50 (Tiny ImageNet)	55.4%	56.52%

Table II. Accuracy of CDLN Compared to Baseline

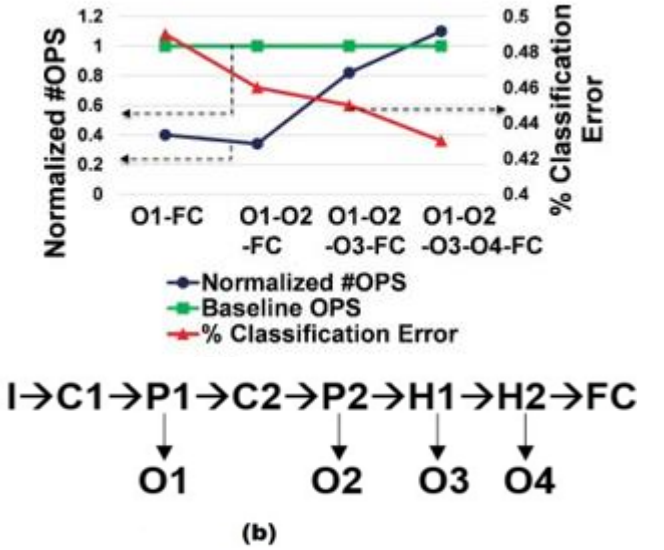
3.2. Improvement in Accuracy

Let's imagine we're in a classroom where the teacher has found a new way to help students (CDL structures) score better on their tests (accuracy) compared to the old teaching method (baseline DLN architecture). The teacher's report (Table II) shows that every student who tried the new method did better than before.

The new method is special because it uses a series of smaller quizzes (linear classifiers) throughout the course, instead of just one big final exam (fully connected output layer). These quizzes are based on what the students have already learned (features from the convolutional layers), and they're easier to study for because they're smaller and more focused. This means students can get really good at these quizzes quickly, leading to better overall scores.

To prove this approach works, the teacher did an experiment with the class (LeNet-5 for MNIST). They introduced these quizzes one by one, and with each new quiz, the students' scores kept getting better. When they added just one quiz, the average score went up a little (from **97.55% to 97.65%**), but when they added three quizzes, the improvement was much bigger (up to **98.92%**). This showed that the more quizzes they took, the fewer mistakes they made, which confirmed that the new teaching method really does help students learn better.

4. Cost & Accuracy Analysis



Let's think of this scenario as a game where players (input instances) go through a series of levels (output layers) in a video game (the DLN trained with ICDL for MNIST). Each level has a certain number of obstacles (operations or computations) they must overcome. The goal is to complete the game with as few obstacles as possible, which means the player is very efficient.

In Figure 4(a), we see a scorecard that shows how players are doing. As they pass through the first two levels (O1-O2-FC), they overcome fewer obstacles, showing a **3.14 times reduction** in the number of obstacles. This is the "break-even point," where the game is most efficient. But after this point, even though there are more obstacles, the players make fewer mistakes (lower error rate), which is good because it means they're getting better at the game.

One interesting point is when we only look at the first level (O1-FC), the players make very few mistakes—only **0.49%**. This is a **7.5% improvement** compared to players who didn't use this new method. It's like they've found a shortcut that makes the game easier to play.

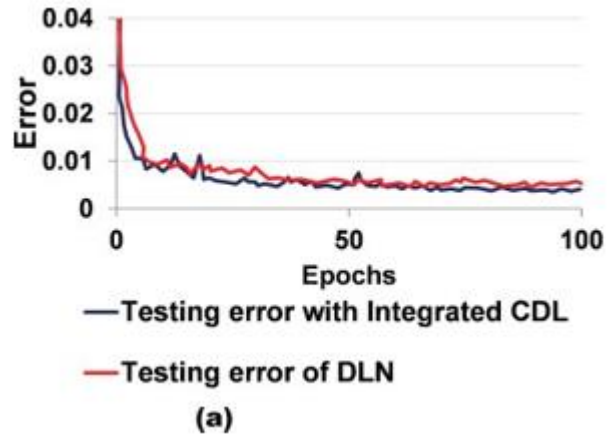
Now, imagine we have two versions of the game. One is the original (standard CDL), and the other is a new and improved version (ICDL). In Figure 4(b), we compare how often players make mistakes in each version. It's clear that the new version helps players make fewer mistakes at every level. Before, in the original game, players would often get stuck early on (misclassified at earlier stages), but with the new version, they're getting past those tricky parts right from the start. This means the new game not only makes players more efficient but also better at each step of the way, proving that the new method (ICDL) really enhances the game's training program, making it more effective overall.

5. Improved Gradient Convergence with ICDL

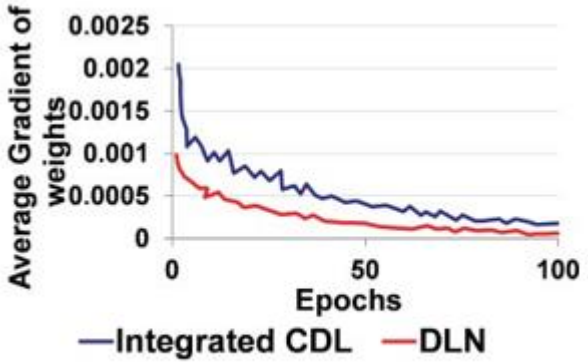
The essence of the integrated training approach is to boost the accuracy of classifications. To achieve the lowest error rates, it's necessary to include additional output stages at every layer of the Deep Learning Network (DLN) during testing. This comprehensive setup, detailed in Figure 12(b) and Table III, goes beyond the break-even point, meaning that while it increases the number of operations per second (#OPS), it doesn't outweigh the cost of the extra layers.

However, despite the increase in #OPS, this full configuration is crucial for achieving the highest possible accuracy during tests. When compared to other advanced DLN methods, the integrated Conditional Deep Learning (CDL) training shows a notable decrease in error rates, dropping to **0.44%** from **0.53%** for MNIST and to **10.76%** from **11.68%** for CIFAR10. This improvement is significant when compared to the standard DLN architectures.

The integrated training not only enhances accuracy by about **1.4–1.5%** for both datasets but also refines the gradient convergence within the DLN. This is a stark contrast to the previous method where a CDL is constructed atop an already trained DLN. The result is a substantial leap in performance with the integrated CDL approach, demonstrating its effectiveness over prior techniques. Essentially, this method fine-tunes the learning process, making intermediate layers more robust and capable of discriminating features, which leads to a more accurate and efficient classification system.



(a)



(b)

6. Comclusion

In the realm of computer vision, deep-learning convolutional networks are essential but require a lot of computational power. This study introduces a new method to make these networks more efficient by using features from the convolutional layers to tell apart simple from complex input data. The idea is called Conditional Deep Learning (CDL), where easy cases are sorted out early on, so the network doesn't have to work as hard. This is done by adding a linear network to each convolutional layer and checking its output to see if we can stop the classification process there. The system adapts the number of layers it uses based on how tough the input is, creating a CDL that's as efficient as possible.

To test out CDL, it was applied to top-notch deep-learning structures for datasets like MNIST, CIFAR10, and Tiny ImageNet. The experiments showed that not only does CDL save energy, but it also makes the networks more accurate compared to the standard models. Taking inspiration from these findings, Integrated CDL training was developed. This method builds a CDL from the ground up by using the mistakes made by the extra linear classifiers during training to improve the network. The end result? The networks trained this way on MNIST and CIFAR10 datasets were significantly more accurate. Essentially, this approach fine-tunes the network's learning process, leading to smarter and more resourceful deep-learning systems.

REFERENCES

Yoshua Bengio. 2009. Learning deep architectures for AI. Found. Trends Mach. Learn. 2, 1 (2009), 1–127.

Dan Ciresan, Ueli Meier, and Ju¨ rgen Schmidhuber. 2012. Multi-column deep neural networks for image classification. In 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 3642–3649.

Ronan Collobert, Koray Kavukcuoglu, and Cle´ment Farabet. 2011. Torch7: A matlab-like environment for machine learning. In BigLearn, NIPS Workshop.

George E. Dahl, Dong Yu, Li Deng, and Alex Acero. 2012. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. IEEE Trans. Audio Speech Lang. Process. 20, 1 (2012), 30–42.

Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V. Le, and others. 2012. Large scale distributed deep networks. In Advances in Neural Information Processing Systems. 1223–1231.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the International Conference on Artificial Intelligence and Statistics. 249– 256.

Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. 2013. Maxout net- works. In Proceedings of The 30th International Conference on Machine Learning. 1319–1327.

Lucas Hansen. 2015. Tiny imagenet challenge submission. CS 231N (2015).

Douglas M. Hawkins. 2004. The problem of overfitting. J. Chem. Inform. Comput. Sci. 44, 1 (2004), 1–12. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015a. Deep residual learning for image recog-

nition. arXiv preprint arXiv:1512.03385 (2015).

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015b. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE International Confer- ence on Computer Vision. 1026–1034.

Geoffrey Hinton, Li Deng, Dong Yu, George E. Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and others. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Sign. Process. Mag. 29, 6 (2012), 82–97.

Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A fast learning algorithm for deep belief nets.

Neur. Comput. 18, 7 (2006), 1527–1554.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580 (2012).

Kevin Jarrett, Koray Kavukcuoglu, Marc'Aurelio Ranzato, and Yann LeCun. 2009. What is the best multi- stage architecture for object recognition? In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision. IEEE, 2146–2153.

Alex Krizhevsky. 2009. Learning multiple layers of features from tiny images. (2009).

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolu- tional neural networks. In Advances in Neural Information Processing Systems. 1097–1105.

Yann LeCun, Le´on Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. Proc. IEEE 86, 11 (1998), 2278–2324.

Yann LeCun, Fu Jie Huang, and Leon Bottou. 2004. Learning methods for generic object recognition with invariance to pose and lighting. In Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04), Vol. 2. IEEE, II–97.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. 2011. Reading digits in natural images with unsupervised feature learning. In Proceedings of the NIPS Workshop on Deep Learning and Unsupervised Feature Learning, Vol. 2011.

Rasmus Berg Palm. 2012. Prediction as a candidate for learning deep hierarchical models of data. Technical University of Denmark 5 (2012).