Journal of Engineering Science
Fascicle *Electronics and Computer Science*
Topic *Computers and Information Technology*
Vol. XXVII, no. 4 (2020), pp. 197 - 207
ISSN 2587-3474
eISSN 2587-3482

DOI: 10.5281/zenodo.4298709
CZU 004.8/.9

OVERVIEW OF COMPUTER VISION SUPERVISED LEARNING TECHNIQUES FOR LOW-DATA TRAINING

Alexandru Burlacu*, ORCID: 0000-0002-8600-5771

Technical University of Moldova, Studenților 9/7, Chișinău, Republic of Moldova
**alexandru.burlacu@faf.utm.md*

Received: 04. 18. 2020
Accepted: 05. 28. 2020

Abstract. In the age of big data and machine learning the costs to turn the data into fuel for the algorithms is prohibitively high. Organizations that can train better models with fewer annotation efforts will have a competitive edge. This work is an overview of techniques of varying complexity and novelty for supervised, or rather weakly supervised learning for computer vision algorithms. The paper starts describing various methods to ease the need for a big labeled dataset with giving some background on supervised, weakly-supervised and then self-supervised learning in general, and in computer vision specifically. The paper describes the importance of these methods in fields such as medical imaging and autonomous driving.

Keywords: *knowledge distillation, knowledge transfer, self-supervised learning, semi-supervised learning, weakly-supervised learning.*

Rezumat. În epoca Big Data și a învățării automate devine tot mai costisitor de a converti datele în combustibil pentru algoritmi. Organizațiile ce sunt capabile de a crea modele algoritmice mai performante ce pot adnota datele cu efort minim față de concurenți au un avantaj competitiv major. Lucrarea dată prezintă un ansamblu de tehnici de complexitate și noutate variabilă pentru algoritmi cu învățare supervizată pentru probleme de vedere artificială. Articolul începe cu descrierea diferitor metode ce pot ușura necesitatea de a avea un set mare de date adnotate prin prezentarea câtorva tehnici bazate pe învățare supervizată, semi-supervizată și auto-supervizată. Ulterior sunt descrise diferite seturi de date tradițional utilizate pentru a estima performanța acestor algoritmi. Pe parcurs articolul descrie importanța acestor metode pentru asemenea domenii ca imagistica medicală și conducere autonomă.

Cuvinte cheie: *distilare de cunoștințe, transfer de cunoștințe, învățare supravegheată, învățare semi-supravegheată, învățare slab supravegheată.*

Introduction

With the advent of deep learning the number of organizations and practitioners who think that they can solve problems using it also grows [1]. Deep learning algorithms normally require vast amounts of labeled data, but depending on the domain it is not always possible to have a well-annotated huge dataset, just think about healthcare.

In recent years deep learning on visual data has become one of the hottest subdomains of machine learning, especially given such well-recognized tasks as the ImageNet Large Scale Visual Recognition Challenge (ILSVRC)[2], PASCAL VOC challenges [3], and KITTI benchmark suite [4]. Image recognition is of paramount importance in medical image analysis, self-driving cars technologies, and other implementations. Deep learning works extremely well on this kind of tasks, but on the flip side, it requires vast amounts of carefully annotated data to work properly. The process of data annotation is at least mundane and slow, but sometimes also very expensive because the costs for one or several subject matter experts can be prohibitively high. Also, depending on who does the annotation, it can be prone to under-representation of groups/classes in the population, leading to biases in the data which are hard to solve afterward.

This is the reason why numerous attempts and techniques have evolved in recent years to try to reduce the data volume requirements for the algorithms. In this work, we will cover the whole spectrum, starting from the simplest techniques like data augmentation or transfer learning, and ending with self-supervised learning. Also, active learning and few-shot learning techniques will be briefly discussed.

Neural networks are a class of machine learning algorithms that are very prone to overfitting, due to them being over-parametrized [5], that is having more parameters than necessary.

As a result, if there isn't enough data to learn from, they will start memorizing, which is undesired.

Starting with simple methods like knowledge transfer the paper describes several knowledge distillation techniques and ends with the latest methods from self- and semi-supervised methods like Unsupervised Data Augmentation (UDA) [6], MixMatch [7], Snorkel [8] and adding synthetic tasks to the learning model, thus touching the multi-task learning problem too.

Methods like few-shot learning and active learning are only briefly described to give a holistic view of the problem. Some prominent datasets and their properties are then described. Finally, the paper is closed with conclusions.

Knowledge Transfer

One of the simplest and oldest techniques is to use so-called transfer learning, a basic type of knowledge transfer between tasks, where an existing, pre-trained model is fine-tuned to be used on downstream tasks (see Figure 1). Transfer learning is considered to work so well because of the hierarchical nature of the learned representations in the neural networks.[9] Starting from simple edge representations and moving on onto more complex shapes, neural networks can re-use a lot of these, therefore needing data only to fine-tune for a given task.

As a rule of thumb, if having more data or the data distribution of the task at hand is not close to the distribution of the pre-training task, for example, the task of interest is medical image analysis and the pre-training task is ImageNet, then more final layers should be retrained, and is recommended to also fine-tune the whole network with a significantly lower learning rate.[10] For smaller datasets or tasks close to the original pre-training task, usually, the last layer is sufficient to retrain.

Transfer learning has been successfully used in medical image analysis [11] and various data science competitions.

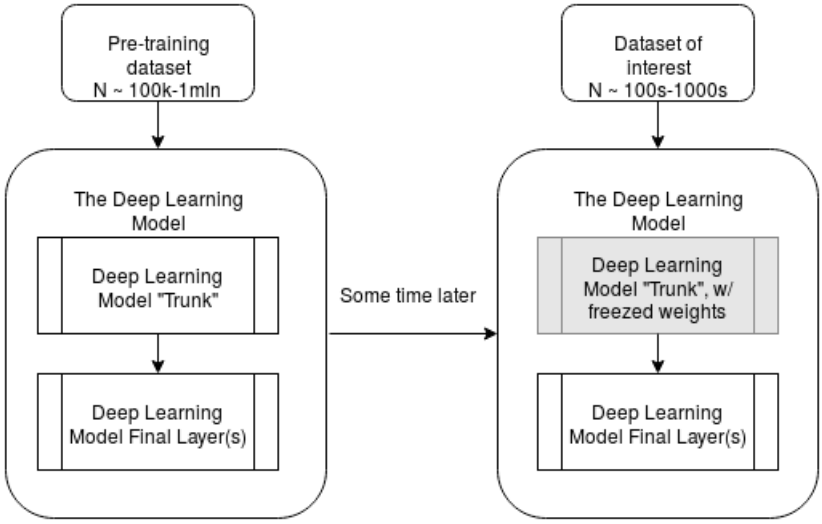

Figure 1. General view of the Transfer Learning process where the left hand side shows the required pre-trained neural network and the right hand side the fine-tuning process.

Data augmentation

Data augmentation is not a tool to solve the lack of data, but it can work as one. It enables us to reuse an existing dataset with a sometimes minor increase of risk of overfitting. In a sufficient data setting, it also acts as a regularizer, by changing the aspect of the data without altering it's meaning, thus making it less likely for the machine learning model to memorize unimportant details about the data. It is, of course, important to remember that some techniques must not be used depending on the original data. Data augmentation must not alter the label(s) of a data point by altering the data point itself.

For computer vision, data augmentation techniques are probably the most developed among modalities. Classic methods are based on rotation, translation and scale manipulations, sometimes shear is used too. Another set of techniques are based on color space manipulation like color inversion, whitening, color shift or changes in brightness (see Figure 2, right). More recently, a transformation method named Cutout [12] was proposed and seem to work fine. Cutout is based on occlusions of images (see Figure 2, left). Also, more and more papers present methods that automatically discover optimal data augmentation procedures for a given problem.

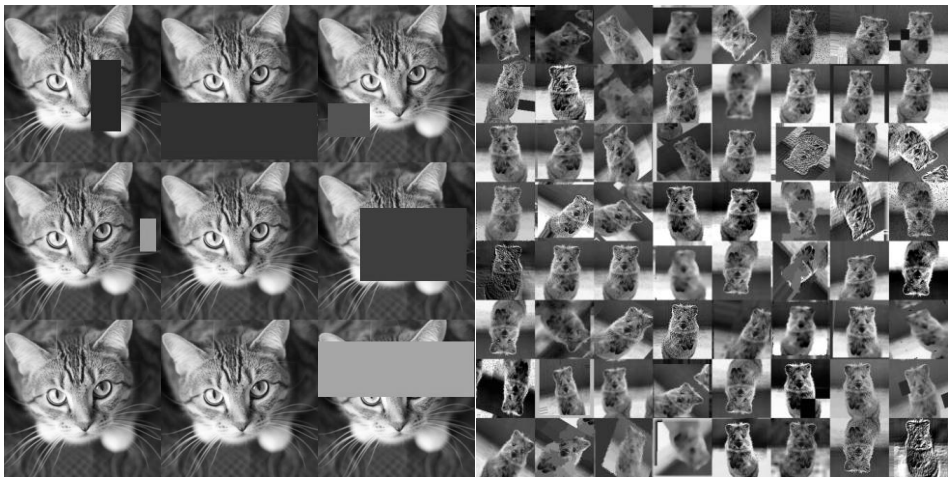

Figure 2. (Left) An example of applying the Cutout augmentation to an image. Source: github.com/xkumiyu/numpy-data-augmentation. (Right) A sample of possible data augmentation schemes. Source: github.com/aleju/imgaug.

Knowledge Distillation

Although this was initially a technique for model compression, and not explicitly designed to alleviate the need for a big dataset, it works by using a pre-trained, teacher model to transfer knowledge into a smaller, student network.[13] Assuming a combination of this technique, lots of unlabeled data, and data augmentation, it is possible to obtain a small yet sufficiently good student network on the final task. This approach is based on the so-called Data Distillation[14]. Also, there seems to be interest in using the teacher model to give noisy labels for unlabeled parts of datasets, thus entering the realm of weakly-supervised and semi-supervised learning [15].

Model Distillation

Model distillation, as a type of knowledge distillation, requires at least one teacher model, which is a bigger model that is already trained and a smaller, student model. The student tries to predict a softened version of the outputs of the bigger model, or ensemble of models. High temperature in softmax is commonly used as a way to soften the predictions from the teacher model (see Python code below).

```
# np stands for numpy linear algebra library
def softmax(x, temperature=1):
        return np.exp(x / temperature) / np.sum(np.exp(x / temperature))
```

By doing so the student model, as stated by Hinton et al, learns not just the true, or mostly true outputs, but also the "dark" knowledge of the teacher model, hidden "representations" encoded into the teacher model during its training.

Data Distillation

Another, more recent way of knowledge distillation is the so-called Data Distillation [14], where instead of an ensemble of teacher models just one would suffice. The additional information is taken from applying augmentations on the available data. In principle, such a model could work even with unlabeled datasets (see Figure 3).

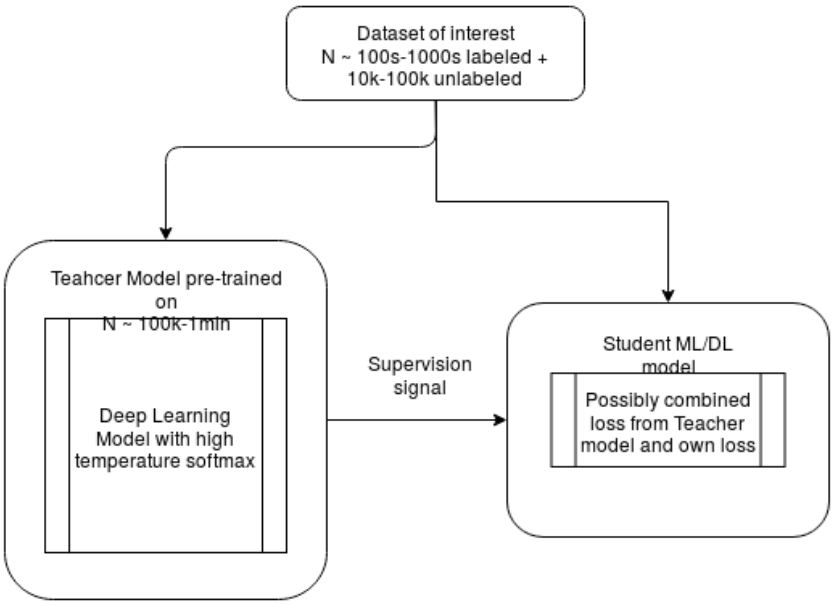

Figure 3. The diagram shows conceptually what components are employed in the Data Distillation [14] process.

Weakly-supervised and Semi-supervised methods

Weakly-supervised learning is an umbrella term for techniques that use datasets that are either partially labeled, commonly known as semi-supervised learning, with noisy labels, known in the literature as inaccurate supervision, or with coarse-grained labels, also known as inexact supervision [16]. A very popular tool called Snorkel can be attributed to the second type of weakly-supervised learning.

In Semi-supervised learning, unlabeled examples provide regularization and a hint to the underlying distribution of the data. There is a couple of assumption that makes it possible to leverage the unlabeled dataset. These are (1) Continuity assumption, (2) Cluster assumption, and (3) Manifold assumption.

Continuity assumption states that points that are close to each other are more likely to share a label. In the case of semi-supervised learning, the smoothness assumption additionally prefers decision boundaries in low-density regions, that is, few points are close to each other but in different classes.

Cluster assumption states that the data tends to form clusters, and points in the same cluster are more likely to share a label. Also, note that data points that share a label may be spread across multiple clusters. This is a special case of the smoothness assumption and gives rise to feature learning with clustering algorithms.

The assumption that data lie approximately on a manifold of much lower dimension than that of the input space is called manifold assumption and is a widespread assumption in machine learning in general. In the case of semi-supervised learning the manifold, based on both the annotated and unlabeled data points, can avoid the curse of dimensionality, because of the possibility to limit the search space of the problem.

The simplest Semi-supervised technique is called Pseudo-Labeling. The core idea is simple. Having a partially annotated dataset, first, train a model using the labeled part of the dataset, then, if it performs fairly well, use it to label the unlabeled part of the dataset. Once done, combine both and retrain the model. Of course, it might be the case that the pseudo-labels are actually wrong, so keep in mind that this is just a heuristic. Anyway, most of the time it works.

Two recent and most prominent solutions for semi-supervised learning are MixMatch [7] and Unsupervised Data Augmentation [6]. These two techniques combine labeled and unlabeled data in a clever way. Namely, both bootstrap part of their training set by classifying unlabeled data and combining loss functions.

Snorkel

Snorkel [8] is a tool developed at Stanford University's DAWN Laboratory, which aims to leverage unlabeled data using low-quality supervision signal based on some heuristics and constraints. That is, to use Snorkel, the user has to provide some functions, probably developed with the help of subject matter experts, that must give a label for a given data point based on some heuristic, rule of thumb. After that, the system aggregates the predictions from these labeling functions and trains a generative model that estimates the correlation and accuracy of these functions and outputs a probabilistic label made of the weighted predictions of labeling functions. It does so by identifying the agreement between functions' outputs. In the end, the system uses probabilistic labels to train a final, discriminative model.

Authors specify that for labeling functions several methods can be used. Not only heuristics but also interactions with knowledge bases and usage of other machine learning models trained on different datasets is possible.

MixMatch

MixMatch [7] is a recent algorithm that combines a number of techniques in order to build a useful prior from unlabeled data such that subsequently, the model will require fewer annotated examples. Stochastic data augmentation is applied to an unlabeled image a number of times, and each augmented image is fed through the classifier. Then, the average of these predictions is "sharpened" by adjusting the distribution's temperature (see Figure 4). Besides, MixMatch uses a regularization technique called MixUp [17] that using convex combinations of both inputs and labels promotes strictly linear behavior between examples. It requires that the model's output for a convex combination of two inputs is close to the convex combination of the output for each individual input.

An example of code for the MixUp regularization, using Pytorch library, is shown below.

```
# y1, y2 should be one-hot vectors
for(x1, y1), (x2, y2) in zip(loader1, loader2):
        lam = numpy.random.beta(alpha, alpha)
        x = Variable(lam*x1 + (1. - lam)*x2)
        y = Variable(lam*y1 + (1. - lam)*y2)
        optimizer.zero_grad()
        loss(net(x), y).backward()
        optimizer.step()
```

Algorithm 1 MixMatch ingests a batch of labeled data $\mathcal{X}$ and a batch of unlabeled data $\mathcal{U}$ and produces a collection $\mathcal{X}'$ of processed labeled examples and a collection $\mathcal{U}'$ of processed unlabeled examples with "guessed" labels.

Input: Batch of labeled examples and their one-hot labels $\mathcal{X} = \big((x_b, p_b); b \in (1, \dots, B)\big)$, batch of unlabeled examples $\mathcal{U} = \big(u_b; b \in (1, \dots, B)\big)$, sharpening temperature T, number of augmentations K, Beta distribution parameter α for MixUp.
for $b = 1$ **to** B **do**
  $\hat{x}_b = \text{Augment}(x_b)$ *// Apply data augmentation to* x_b
  for $k = 1$ **to** K **do**
    $\hat{u}_{b,k} = \text{Augment}(u_b)$ *// Apply* k^{th} *round of data augmentation to* u_b
  end for
  $\bar{q}_b = \frac{1}{K} \sum_k \text{p}_{\text{model}}(y \mid \hat{u}_{b,k}; \theta)$ *// Compute average predictions across all augmentations of* u_b
  $q_b = \text{Sharpen}(\bar{q}_b, T)$ *// Apply temperature sharpening to the average prediction (see eq. (7))*
end for
$\hat{\mathcal{X}} = \big((\hat{x}_b, p_b); b \in (1, \dots, B)\big)$ *// Augmented labeled examples and their labels*
$\hat{\mathcal{U}} = \big((\hat{u}_{b,k}, q_b); b \in (1, \dots, B), k \in (1, \dots, K)\big)$ *// Augmented unlabeled examples, guessed labels*
$\mathcal{W} = \text{Shuffle}\big(\text{Concat}(\hat{\mathcal{X}}, \hat{\mathcal{U}})\big)$ *// Combine and shuffle labeled and unlabeled data*
$\mathcal{X}' = \big(\text{MixUp}(\hat{\mathcal{X}}_i, \mathcal{W}_i); i \in (1, \dots, |\hat{\mathcal{X}}|)\big)$ *// Apply* MixUp *to labeled data and entries from* $\mathcal{W}$
$\mathcal{U}' = \big(\text{MixUp}(\hat{\mathcal{U}}_i, \mathcal{W}_{i+|\hat{\mathcal{X}}|}); i \in (1, \dots, |\hat{\mathcal{U}}|)\big)$ *// Apply* MixUp *to unlabeled data and the rest of* $\mathcal{W}$
return $\mathcal{X}', \mathcal{U}'$

Figure 4. The MixMatch algorithm as described in the original paper [7].

Unsupervised Data Augmentation

Another recent approach towards efficient learning with little annotated data comes from Google and is called Unsupervised Data Augmentation, UDA [6] for short. UDA, just like MixMatch uses data augmentation to its advantage. In the case of UDA, a combined loss of cross-entropy, for labeled data, and a consistency loss applied to a pair of the original unlabeled example, and its randomly augmented variant (see Figure 5). The consistency loss is based on the KL-divergence. This scheme proves to be efficient in using the unlabeled part of the dataset. UDA applies simpler data augmentation schemes to the

labeled examples too. In the original paper, authors apply UDA not only to vision problems but also to text, consistently achieving superior results.

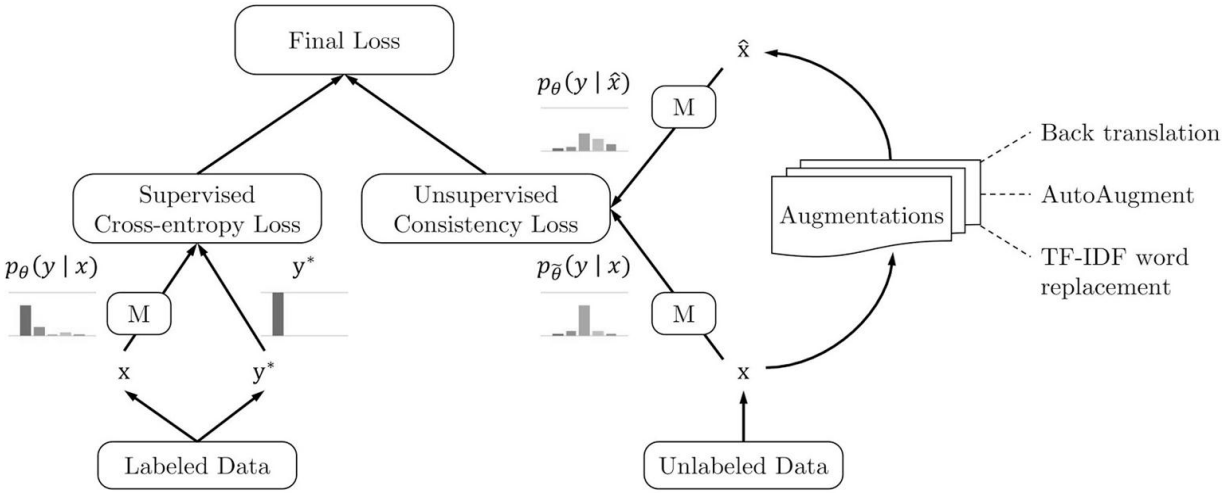

Figure 5. The description of the UDA technique as of the original paper [6].

Self-supervised learning

It is possible to bootstrap a supervision signal out of an unlabeled dataset by trying to predict parts of it, or some features for example. This approach is called self-supervised learning.

Thus without explicit labels self-supervised learning makes a seemingly unsupervised learning problem into a supervised one. For example, the prediction of affine transformations applied on an image is a technique initially regarded as unsupervised pre-training [18], and it's authors argued that by doing so, the model is forced to learn only features that vary slightly. Such features are significant and can be easily used to discriminate between classes. More recently, such tasks as solving a jigsaw puzzle, coloring, filling of cropped regions are used for self-supervised learning for neural networks acting on image input.[19] In case of a video signal common methods are trying to order frames or learn the optical flow in some way.

A robust way to ensure an even stronger supervision signal for a task with little labeled data is to combine tasks during training [19], thus forcing a model to learn features that are transferable between tasks and therefore significant for the classes in the dataset. For example, combining several self-supervised tasks with the main classification task.

Datasets

Caltech 256 dataset

Released in 2006, the Caltech 256 dataset [20], by Griffin, Holub, and Perona, is an image-based dataset consisting of more than 30000 images of 256 classes. It is considered an improvement to its predecessor, the 2003 Caltech 101 dataset, due to the increased number of classes and more cluttered images, making it harder to learn. With the number of images per class ranging between 80 and more than 800 images, with 119 on average it proves to be quite challenging.

CIFAR datasets

The CIFAR-10 and CIFAR-100 [21] are labeled subsets of the 80 million tiny images dataset. They were collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. They are

almost an omnipresent benchmark for computer vision deep learning algorithms. Both CIFAR-10 and CIFAR-100 have 60000 images, 50000 for training and 10000 for validation, and they differ only by the number of available classes. CIFAR-100 is a refinement of CIFAR-10 with those 100 classes being grouped into 20 superclasses. Each image comes with a "fine" label (the class to which it belongs) and a "coarse" label (the superclass to which it belongs).

PASCAL VOC 2007

PASCAL VOC [3] datasets are primarily used for object detection and segmentation tasks. The 2007 dataset is made up of 20 classes and the data has been split into 50% for training/validation and 50% for testing. The distributions of images and objects by class are approximately equal across the training/validation and test sets. In total there are almost 10000 images, containing almost 25000 annotated objects.

STL-10 dataset

The STL-10 [22] dataset's primary applications, outlined by its authors, are for developing unsupervised feature learning, deep learning, and self-taught learning algorithms. It based on ImageNet dataset, and is inspired by the CIFAR-10, namely has 10 classes but with only 500 training labeled images, and 800 more test images (see Figure 6). A set of 100000 unlabeled examples is provided to learn image models prior to supervised training. These are extracted from a similar but broader distribution of images than the base dataset. It contains other types of related classes in addition to the ones in the labeled set. The primary challenge posed by the authors is to make use of the unlabeled data, with a similar but different distribution, to build a useful prior. By increasing the resolution of the examples to 96x96 pixels, they also expect it to be a challenging benchmark for developing more scalable unsupervised learning techniques.

Figure 6. A sample of the STL-10 dataset [22] representing images for all ten classes in it.

ImageNet dataset

The ImageNet project [2], first presented by Fei-Fei Li in 2009 at CVPR, is a large visual database which design was inspired by the WordNet taxonomy, for use in visual object recognition software research. It is made of more than 14 million images, which were hand-annotated using Amazon Mechanical Turk service, and for at least 1000000 images, bounding boxes are also provided. ImageNet contains more than 20,000 categories, out of which 1000 non-overlapping categories are part of the trimmed down to 1000000 images dataset used since 2010 in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) competition. The deep learning saw an upheaval starting 2012 when the AlexNet model won ILSVRC with a 10.8% error difference with the runner up.

Caltech-UCSD Birds (CUB) 200 dataset

Especially popular in the active learning community, the CUB-200 [23], the 2011 and 2010 editions, the dataset is a very challenging one due to the closeness of the classes in it. It is made of about 7000 and 12000 images for 2010 and 2011 years versions respectively and consists of 200 different species of birds (see Figure 7), mostly from North America.

Figure 7. A sample of the CUB-200-2011 dataset [23].

iNaturalist 2018 dataset

In recent works on semi-supervised and self-supervised learning, a new dataset has been used, the dataset from the iNaturalist 2018 Kaggle challenge. The iNaturalist Species Classification and Detection Dataset's [24] authors argue that to advance the computer vision discipline we need datasets that are more aligned with the real-world conditions. Their approach consists of collecting over 850000 images of plants and animals, from over 5000 categories, made with different types of cameras and in different illumination conditions. Moreover, there is a big class imbalance present in the dataset which drastically affects existing neural networks results on it. The authors observed poor results specifically for classes with small numbers of training examples, suggesting more attention is needed in low-shot learning to achieve decent results on this dataset.

Final words on datasets

In general, for low-shot learning, we need specially designed datasets. Of course, a small number of images per class is necessary, but to truly test algorithms capabilities, we need these images to have varying quality and made in different real-world conditions.

Also, having an ImageNet like dataset, with a well-ordered taxonomy, could be further used in tasks requiring efficient knowledge transfer between tasks, another important direction towards true AI.

Active learning

Active learning means having a human in the loop that will interactively label the data points classified with low confidence by some model. The assumption for this kind of learning is that fixing low certainty data points can quickly increase the predictive power of a classifier. It's a rather handy way to bootstrap a dataset. For a good overview of the technique see [25].

Few-shot learning

Few-shot learning is posed as a problem of how to learn a class from very few examples, sometimes even from one example. It is mostly a problem of model parameter initialization, a meta-learning approach, or as a problem of matching classes/instances. The field is broad and existing techniques are rarely used in practice (yet). For a survey on various approaches within the few-shot learning technique see [26].

Conclusions

Most of the time a combination of techniques works best. Also, one should always keep in mind that some methods cannot be used due to problem constraints, for example in case of data augmentation. Most techniques outlined above still require a lot of data, but this time unlabeled, thus alleviating the expenses and time requirements to label it. Some other problems are still out there, mostly regarding privacy or representability.

No method outlined above can work in extremely low data setups, like a few samples per class. They all require at least tens to hundreds of labeled samples, depending on the hardness of the problem. For this kind of problem, few-shot learning is the only way. This paper does not go into few-shot and one-shot learning, because this is another huge sub-domain, with a scope a bit different from the one of weakly-supervised and self-supervised learning.

The general trends in empowering algorithms to use less labeled data are multi-task learning, self-supervision and relying on existing representations. Combining these three methods it is possible to design machine learning systems that learn efficiently and give a competitive edge.

Another problem needed to be solved to accelerate the development of new data-efficient algorithms is the lack of proper benchmarks and datasets. Of course, there is STL-10 and CUB-200 datasets, but this is not enough. The new iNaturalist dataset is a move in the right direction concerning the design of datasets closely resembling real-world scenarios.

Acknowledgments. The work was approved at the International Conference on Electronics, Communications and Computing, ECCO – 2019.

References

1. Technavio. *Global Deep Learning System Market 2016-2020*. Report. 2016.
2. Deng J., Dong W. et al. *ImageNet: A Large-Scale Hierarchical Image Database*. IEEE Computer Vision and Pattern Recognition (CVPR). 2009.
3. Everingham M., Van Gool L., et al. *The {PASCAL} {V}isual {O}bject {C}lasses {C}hallenge 2007 {(VOC2007)} Results,* 2007.

4. Geiger A., Lenz P., et al. *Vision meets Robotics: The KITTI Dataset*. International Journal of Robotics Research (IJRR), 2013.
5. Zeyuan A. Z., Yuanzhi L., Yingyu L. *Learning and Generalization in Overparameterized Neural Networks, Going Beyond Two Layers*. Conference on Neural Information Processing Systems (NeurIPS), 2019.
6. Qizhe X., et al. *Unsupervised Data Augmentation for Consistency Training*. In: *pre-print arXiv:1904.12848*, 2019.
7. Berthelot D., et al. *MixMatch: A Holistic Approach to Semi-Supervised Learning*. In: *pre-print arXiv:1905.02249*, 2019.
8. Ratner A., et al. *Snorkel: Rapid Training Data Creation with Weak Supervision*. In: *44th International Conference on Very Large Data Bases,* 2018.
9. Chollet F. *Building powerful image classification models using very little data* [online]. 2016. [Accessed: 15.09.2019]. Available: https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html
10. Chuanqi T., et al. *A Survey on Deep Transfer Learning*. In: *The 27th International Conference on Artificial Neural Networks (ICANN),* 2018.
11. Raghu M., Zhang C., Kleinberg J., Bengio S. *Transfusion: Understanding Transfer Learning for Medical Imaging*. In: *pre-print* a*rXiv: 1902.07208*, 2019.
12. Devries T., taylor G. *Improved Regularization of Convolutional Neural Networks with Cutout*, In: *pre-print* a*rxiv: 1708.04552*, 2017.
13. Hinton G., Vinyals O., Dean J. *Distilling the Knowledge in a Neural Network*. Deep Learning Workshop at the Conference on Neural Information Processing Systems (NIPS), 2014.
14. Radosavovic I., et al. *Data Distillation: Towards Omni-Supervised Learning*. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
15. Zeki Yalniz I., et al. Billion-scale semi-supervised learning for image classification. In: *pre-print* a*rXiv: 1905.00546*, 2019.
16. Zhou Z. H. *A brief introduction to weakly supervised learning*. In: *National Science Review*, Volume 5, Issue 1, January 2018, Pages 44–53, https://doi.org/10.1093/nsr/nwx106
17. Zhang H., Cisse M., Dauphin Y., Lopez-Paz D. *mixup: Beyond Empirical Risk Minimization*. International Conference on Learning Representations (ICLR), 2017.
18. Agrawal P., Carreira J., Malik J. *Learning to See by Moving*. IEEE International Conference on Computer Vision (ICCV), 2015.
19. Zisserman A. *Self-supervised Learning*. EEML 2019 Summer School, Bucharest, 2019.
20. Griffin G., Holub A., Perona P. *Caltech-256 Object Category Dataset*. CalTech Report, 2007.
21. Coates A., Lee H., Ng A. Y. *An Analysis of Single Layer Networks in Unsupervised Feature Learning*. International Conference on Artificial Intelligence and Statistics (AISTATS), 2011.
22. Krizhevsky A. *Learning Multiple Layers of Features from Tiny Images*. Technical Report, 2009.
23. Wah C., Branson S., Welinder P., Perona P., Belongie S. *The Caltech-UCSD Birds-200-2011 Dataset*. California Institute of Technology, 2011.
24. Van Horn G. et al. *The iNaturalist Species Classification and Detection Dataset*. IEEE Conference on Computer Vision and Pattern Recognition, 2018.
25. Budd S., Robinson E., Kainz B. *A Survey on Active Learning and Human-in-the-Loop Deep Learning for Medical Image Analysis*. In: *pre-print* a*rXiv:1910.02923*, 2019.
26. *Wang Y., Yao Q., Kwok J.T., Ni L.M. Generalizing from a Few Examples: A Survey on Few-Shot Learning. In: pre-print arXiv: 1904.05046, 2019.*