Johnson and Khoshgoftaar *J Big Data* (2019) 6:27
https://doi.org/10.1186/s40537-019-0192-5

Journal of Big Data

SURVEY PAPER Open Access

Survey on deep learning with class imbalance

Justin M. Johnson* and Taghi M. Khoshgoftaar

*Correspondence:
jjohn273@fau.edu
Florida Atlantic University,
777 Glades Road, Boca Raton,
FL, USA

Abstract

The purpose of this study is to examine existing deep learning techniques for addressing class imbalanced data. Effective classification with imbalanced data is an important area of research, as high class imbalance is naturally inherent in many real-world applications, e.g., fraud detection and cancer detection. Moreover, highly imbalanced data poses added difficulty, as most learners will exhibit bias towards the majority class, and in extreme cases, may ignore the minority class altogether. Class imbalance has been studied thoroughly over the last two decades using traditional machine learning models, i.e. non-deep learning. Despite recent advances in deep learning, along with its increasing popularity, very little empirical work in the area of deep learning with class imbalance exists. Having achieved record-breaking performance results in several complex domains, investigating the use of deep neural networks for problems containing high levels of class imbalance is of great interest. Available studies regarding class imbalance and deep learning are surveyed in order to better understand the efficacy of deep learning when applied to class imbalanced data. This survey discusses the implementation details and experimental results for each study, and offers additional insight into their strengths and weaknesses. Several areas of focus include: data complexity, architectures tested, performance interpretation, ease of use, big data application, and generalization to other domains. We have found that research in this area is very limited, that most existing work focuses on computer vision tasks with convolutional neural networks, and that the effects of big data are rarely considered. Several traditional methods for class imbalance, e.g. data sampling and cost-sensitive learning, prove to be applicable in deep learning, while more advanced methods that exploit neural network feature learning abilities show promising results. The survey concludes with a discussion that highlights various gaps in deep learning from class imbalanced data for the purpose of guiding future research.

Keywords: Deep learning, Deep neural networks, Class imbalance, Big data

Introduction

Supervised learning methods require labeled training data, and in classification problems each data sample belongs to a known class, or category [1, 2]. In a binary classification problem with data samples from two groups, class imbalance occurs when one class, the minority group, contains significantly fewer samples than the other class, the majority group. In many problems [3–7], the minority group is the class of interest, i.e., the positive class. A well-known class imbalanced machine learning scenario is the medical diagnosis task of detecting disease, where the majority of the patients are

healthy and detecting disease is of greater interest. In this example, the majority group of healthy patients is referred to as the negative class. Learning from these imbalanced data sets can be very difficult, especially when working with big data [8, 9], and non-standard machine learning methods are often required to achieve desirable results. A thorough understanding of the class imbalance problem and the methods available for addressing it is indispensible, as such skewed data exists in many real-world applications.

When class imbalance exists within training data, learners will typically over-classify the majority group due to its increased prior probability. As a result, the instances belonging to the minority group are misclassified more often than those belonging to the majority group. Additional issues that arise when training neural networks with imbalanced data will be discussed in the "Deep learning methods for class imbalanced data" section. These negative effects make it very difficult to accomplish the typical objective of accurately predicting the positive class of interest. Furthermore, some evaluation metrics, such as accuracy, may mislead the analyst with high scores that incorrectly indicate good performance. Given a binary data set with a positive class distribution of 1%, a naïve learner that always outputs the negative class label for all inputs will achieve 99% accuracy. Many traditional machine learning techniques, which are summarized in the "Machine learning techniques for class imbalanced data" section, have been developed over the years to combat these adverse effects.

Methods for handling class imbalance in machine learning can be grouped into three categories: data-level techniques, algorithm-level methods, and hybrid approaches [10]. Data-level techniques attempt to reduce the level of imbalance through various data sampling methods. Algorithm-level methods for handling class imbalance, commonly implemented with a weight or cost schema, include modifying the underlying learner or its output in order to reduce bias towards the majority group. Finally, hybrid systems strategically combine both sampling and algorithmic methods [10].

Over the last 10 years, deep learning methods have grown in popularity as they have improved the state-of-the-art in speech recognition, computer vision, and other domains [11]. Their recent success can be attributed to an increased availability of data, improvements in hardware and software [12–16], and various algorithmic breakthroughs that speed up training and improve generalization to new data [17]. Despite these advances, very little statistical work has been done which properly evaluates techniques for handling class imbalance using deep learning and their corresponding architectures, i.e. deep neural networks (DNNs). In fact, many researchers agree that the subject of deep learning with class imbalanced data is understudied [18–23]. For this reason, our survey is limited to just 15 deep learning methods for addressing class imbalance.

A comprehensive literature review was performed in order to identify a broad range of deep learning methods for addressing class imbalance. We have documented the specific details of the literature search process so that other scholars may more confidently use this survey in future research, an essential step in any literature review [24]. Candidate papers were first discovered through the Google Scholar [25] and IEEE Xplore [26] databases. Keyword searches included combinations of query terms such as: “class imbalance”, “class rarity”, “skewed data”, “deep learning”, “neural networks” and “deep neural networks”. Search results were reviewed and filtered, removing those that did

not demonstrate learning from class imbalanced data with neural networks containing two or more hidden layers. No restrictions were placed on the date of publication. The matched search results were then used to perform backward and forward searches, i.e. reviewing the references of matched articles and additional sources that have cited these articles. This was repeated until all relevant papers were identified, to the best of our knowledge.

Additional selection criteria were applied to exclude papers that only tested low levels of class imbalance, that did not compare proposed methods to other existing class imbalance methods, or that only used a single data set for evaluation. We discovered that papers meeting these criteria are very limited. Therefore, in order to increase the total number of selected works, these additional requirements were relaxed. The final set of 15 publications includes journal articles, conference papers, and student theses that employ deep learning methods with class imbalanced data.

We explore a variety of data-level, algorithm-level, and hybrid deep learning methods designed to improve the classification of imbalanced data. Implementation details, experimental results, data set details, network topologies, class imbalance levels, performance metrics, and any known limitations are included in each surveyed work's discussion. Tables 17 and 18, in the "Discussion of surveyed works" section, summarize all of the surveyed deep learning methods and the details of their corresponding data sets. This survey provides the most current analysis of deep learning methods for addressing class imbalance, summarizing and comparing all related work to date, to the best of our knowledge.

The remainder of this paper is organized as follows. The "Class imbalance background" section provides background information on the class imbalance problem, reviews performance metrics that are more sensitive to class imbalanced data, and discusses some of the more popular traditional machine learning (non-deep learning) techniques for handling imbalanced data. The "Deep learning background" section provides necessary background information on deep learning. The neural network architectures used throughout the survey are introduced, along with several important milestones and the use of deep learning in solving big data analytics challenges. The "Deep learning methods for class imbalanced data" section surveys 15 published studies that analyze deep learning methods for addressing class imbalance. The "Discussion of surveyed works" section summarizes the surveyed works and offers further insight into their various strengths and weaknesses. The "Conclusion" section concludes the survey and discusses potential areas for future work.

Class imbalance background

The task of binary classification, comprised of one positive group and one negative group, is used to discuss class imbalance and various techniques for addressing its challenges in this section. These concepts can be extended to the multi-class problem, because it is possible to convert multi-class problems into a set of two-class problems through class decomposition [27].

The class imbalance problem

Skewed data distributions naturally arise in many applications where the positive class occurs with reduced frequency, including data found in disease diagnosis [3], fraud detection [4, 5], computer security [6], and image recognition [7]. *Intrinsic imbalance* is the result of naturally occurring frequencies of data, e.g. medical diagnoses where the majority of patients are healthy. *Extrinsic imbalance*, on the other hand, is introduced through external factors, e.g. collection or storage procedures [28].

It is important to consider the representation of the minority and majority classes when learning from imbalanced data. It was suggested by Krawczyk [10] that good results can be obtained, regardless of class disproportion, if both groups are well represented and come from non-overlapping distributions. Japkowicz [29] examined the effects of class imbalance by creating artificial data sets with various combinations of complexity, training set size, and degrees of imbalance. The results show that sensitivity to imbalance increases as problem complexity increases, and that non-complex, linearly separable problems are unaffected by all levels of class imbalance.

In some domains, there is a genuine lack of data due to the low frequency with which events occur, e.g. detecting oil spills [7]. Learning from extreme class imbalanced data, where the minority class accounts for as few as 0.1% of the training data [10, 30], is of great importance because it is typically these rare occurrences that we are most interested in. Weiss [31] discusses the difficulties of learning from *rare events* and various machine learning techniques for addressing these challenges.

The total number of minority samples available is of greater interest than the ratio or percentage of the minority. Consider a minority group that is just 1% of a data set containing 1 million samples. Regardless of the high level of imbalance, there are still many positive samples (10,000) available to train a model. On the other hand, an imbalanced data set where the minority class displays rarity or under-representation is more likely to compromise the performance of the classifier [30].

$$\rho = \frac{max_i\{|C_i|\}}{min_i\{|C_i|\}} \tag{1}$$

For the purpose of comparing experimental results across all works presented in this survey, a ratio ρ (Eq. 1) [23] will be used to indicate the maximum between-class imbalance level. C_i is a set of examples in class i, and $max_i\{|C_i|\}$ and $min_i\{|C_i|\}$ return the maximum and minimum class size over all i classes, respectively. For example, if a data set's largest class has 100 samples and its smallest class has 10 samples, then the data has an imbalance ratio of $\rho = 10$. Since the actual number of samples may prove more important than the ratio, Table 18 also includes the maximum and minimum class sizes for all experiments in this survey.

Performance metrics

The confusion matrix in Table 1 summarizes binary classification results. The FP and FN errors correspond to Type I and Type II errors, respectively. All of the performance metrics listed in this section can be derived from the confusion matrix.

Table 1 Confusion matrix

	Actual positive	Actual negative
Predicted positive	True positive (TP)	False positive (FP)
Predicted negative	False negative (FN)	True negative (TN)

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

$$Error\ Rate = 1 - Accuracy \tag{3}$$

Accuracy (Eq. 2) and *error rate* (Eq. 3) are the most frequently used metrics when evaluating classification results. When working with class imbalance, however, both are insufficient, as the resulting value is dominated by the majority group, i.e. the negative class. As mentioned previously, when given a data set whose positive group distribution is just 1% of the data set, a naïve classifier can achieve a 99% accuracy score by simply labeling all examples as negative. Of course, such a model would provide no real value. For this reason, we review several popular evaluation metrics commonly used with imbalanced data problems.

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = TPR = \frac{TP}{TP + FN} \tag{5}$$

$$Selectivity = TNR = \frac{TN}{TN + FP} \tag{6}$$

Precision (Eq. 4) measures the percentage of the positively labeled samples that are actually positive. Precision is sensitive to class imbalance because it considers the number of negative samples incorrectly labeled as positive. Precision alone is insufficient, however, because it provides no insight into the number of samples from the positive group that were mislabeled as negative. On the other hand, *Recall* (Eq. 5), or the *True Positive Rate* (TPR), measures the percentage of the positive group that was correctly predicted to be positive by the model. Recall is not affected by imbalance because it is only dependent on the positive group. Recall does not consider the number of negative samples that are misclassified as positive, which can be problematic in problems containing class imbalanced data with many negative samples. There is a trade-off between precision and recall, and the metric of greater importance varies from problem to problem. *Selectivity* (Eq. 6), or the *True Negative Rate* (TNR), measures the percentage of the negative group that was correctly predicted to be negative.

$$F\text{-}Measure = \frac{(1 + \beta^2) \times Recall \times Precision}{\beta^2 \times Recall + Precision} \tag{7}$$

$$G\text{-}Mean = \sqrt{TPR \times TNR} \tag{8}$$

$$Balanced\ Accuracy = \frac{1}{2} \times (TPR + TNR) \tag{9}$$

The *F-Measure* (Eq. 7), or *F1-score*, combines precision and recall using the harmonic mean, where coefficient β is used to adjust the relative importance of precision versus recall. The *G-Mean* (Eq. 8) measures performance by combining both the TPR and the TNR metrics using the square root of their product. Similar to the G-Mean, the *Balanced Accuracy* (Eq. 9) metric also combines TPR and TNR values to compute a metric that is more sensitive to the minority group [18]. Although F-Measure, G-Mean, and Balanced Accuracy are improvements over Accuracy and Error Rate, they are still not entirely effective when comparing performance between classifiers and various distributions [28].

The *receiver operating characteristics (ROC) curve*, first presented by Provost and Fawcett [32], is another popular assessment which plots true positive rate over false positive rate, creating a visualization that depicts the trade-off between correctly classified positive samples and incorrectly classified negative samples. For models which produce continuous probabilities, thresholding can be used to create a series of points along ROC space [28]. From this a single summary metric, the *area under the ROC curve (AUC)*, can be computed and is often used to compare performance between models. A *weighted-AUC* metric, which takes cost biases into consideration when calculating the area, was introduced by Weng and Poon [33].

According to Davis and Goadrich [34], ROC curves can present overly optimistic results on highly skewed data sets and *Precision–Recall (PR) curves* should be used instead. The authors claim that a curve can only dominate in ROC space if it also dominates in PR space. This is justified by the fact that the false positive rate used by ROC, $FPR = \frac{FP}{FP+TN}$, will be less sensitive to changes in FP as the size of the negative class grows.

According to Seliya et al. [35], learners should be evaluated with a set of complementary performance metrics, where each individual metric captures a different aspect of performance. In their comprehensive study, 22 different performance metrics were used to evaluate two classifiers across 35 unique data sets. Common factor analysis was then used to group the metrics, identifying sets of unrelated performance metrics that can be used in tandem to reduce redundancy and improve performance interpretation. One example set of complementary performance metrics discovered by Seliya et al. is AUC, *Brier Inaccuracy* [36], and accuracy.

Machine learning techniques for class imbalanced data

Addressing class imbalance with traditional machine learning techniques has been studied extensively over the last two decades. The bias towards the majority class can be alleviated by altering the training data to decrease imbalance, or by modifying the model's underlying learning or decision process to increase sensitivity towards the minority group. As such, methods for handling class imbalance are grouped into data-level techniques, algorithm-level methods, and hybrid approaches. This section summarizes some of the more popular traditional machine learning methods for handling class imbalance.

Data-level methods

Data-level methods for addressing class imbalance include over-sampling and under-sampling. These techniques modify the training distributions in order to decrease the level of imbalance or reduce noise, e.g. mislabeled samples or anomalies. In their simplest forms, random under-sampling (RUS) discards random samples from the majority group, while random over-sampling (ROS) duplicates random samples from the minority group [37].

Under-sampling voluntarily discards data, reducing the total amount of information the model has to learn from. Over-sampling will cause an increased training time due to the increased size of the training set, and has also been shown to cause over-fitting [38]. Over-fitting, characterized by high variance, occurs when a model fits too closely to the training data and is then unable to generalize to new data. A variety of intelligent sampling methods have been developed in an attempt to balance these trade-offs.

Intelligent under-sampling methods aim to preserve valuable information for learning. Zhang and Mani [39] present several *Near-Miss* algorithms that use a K-nearest neighbors (K-NN) classifier to select majority samples for removal based on their distance from minority samples. *One-sided selection* was proposed by Kubat and Matwin [40] as a method for removing noisy and redundant samples from the majority class as they are discovered through a 1-NN rule and Tomek links [41]. Barandela et al. [42] use *Wilson's editing* [43], a K-NN rule that removes misclassified samples from the training set, to remove majority samples from class boundaries.

A number of informed over-sampling techniques have also been developed to strengthen class boundaries, reduce over-fitting, and improve discrimination. Chawla et al. [44] introduced the *Synthetic Minority Over-sampling Technique* (SMOTE), a method that produces artificial minority samples by interpolating between existing minority samples and their nearest minority neighbors. Several variants to SMOTE, e.g. *Borderline-SMOTE* [45] and *Safe-Level-SMOTE* [46], improve upon the original algorithm by also taking majority class neighbors into consideration. Borderline-SMOTE limits over-sampling to the samples near class borders, while Safe-Level-SMOTE defines safe regions to prevent over-sampling in overlapping or noise regions.

Supervised learning systems usually define a concept with several disjuncts, where each disjunct is a conjunctive definition describing a subconcept [47]. The size of a disjunct corresponds to the number of samples that the disjunct correctly classifies. Small disjuncts, often corresponding to rare cases in the domain, are learned concepts that correctly classify only a few data samples. These small disjuncts are problematic, as they often contain much higher error rates than large disjuncts, and they cannot be removed without compromising performance [48].

Jo and Japkowicz [49] proposed *cluster-based over-sampling* to address the presence of small disjuncts in the training data. Minority and majority groups are first clustered using the K-means algorithm, then over-sampling is applied to each cluster separately. This improves both within-class imbalance and between-class imbalance.

Van Hulse et al. [37] compared seven different sampling techniques with 11 commonly-used machine learning algorithms. Each model was evaluated with 35 benchmark data sets using six different performance metrics to compare results. It was shown that sampling results were highly dependent on both the learner and the evaluation

performance metric. Experiments revealed that RUS resulted in good performance overall, outperforming ROS and intelligent sampling methods in most cases. The results suggest that, although RUS performs well in most cases, no sampling method is guaranteed to perform best in all problem domains, and multiple performance metrics should be used when evaluating results.

Algorithm-level methods

Unlike data sampling methods, algorithmic methods for handling class imbalance do not alter the training data distribution. Instead, the learning or decision process is adjusted in a way that increases the importance of the positive class. Most commonly, algorithms are modified to take a class penalty or weight into consideration, or the decision threshold is shifted in a way that reduces bias towards the negative class.

In cost-sensitive learning, penalties are assigned to each class through a cost matrix. Increasing the cost of the minority group is equivalent to increasing its importance, decreasing the likelihood that the learner will incorrectly classify instances from this group [10]. The cost matrix of a binary classification problem is shown in Table 2 [50]. A given entry of the table, c_{ij}, is the cost associated with predicting class i when the true class is j. Usually, the diagonal of the cost matrix, where $i = j$, is set to 0. The costs corresponding to false positive and false negative errors are then adjusted for desired results.

Ling and Sheng [51] categorize cost-sensitive methods as either a direct method, or a meta-learning method. Direct methods are methods that have cost-sensitive capabilities within themselves, achieved through modification of the learner's underlying algorithm such that costs are taken into consideration during learning. The optimization process changes from one of minimizing total error, to one of minimizing total cost. Meta-learning methods utilize a wrapper to convert cost-insensitive learners into cost-sensitive systems. If a cost-insensitive classifier produces posterior probability estimates, the cost matrix can be used to define a new threshold p^* such that:

$$p^* = \frac{c_{10}}{c_{10} + c_{01}} \tag{10}$$

Usually, thresholding methods use p^* (Eq. 10) to redefine the output decision threshold when classifying samples [51]. Threshold moving, or post-processing the output class probabilities using Eq. 10, is one meta-learning approach that converts a cost-insensitive learner into a cost-sensitive system.

One of the biggest challenges in cost-sensitive learning is the assignment of an effective cost matrix. The cost matrix can be defined empirically, based on past experiences, or a domain expert with knowledge of the problem can define them. Alternatively, the false negative cost can be set to a fixed value while the false positive cost is varied, using a validation set to identify the ideal cost matrix. The latter has the advantage of exploring

Table 2 Cost matrix

	Actual positive	Actual negative
Predicted positive	$C(1,1) = c_{11}$	$C(1,0) = c_{10}$
Predicted negative	$C(0,1) = c_{01}$	$C(0,0) = c_{00}$

a range of costs, but can be expensive and even impractical if the size of the data set or number of features is too large.

Hybrid methods

Data-level and algorithm-level methods have been combined in various ways and applied to class imbalance problems [10]. One strategy includes performing data sampling to reduce class noise and imbalance, and then applying cost-sensitive learning or thresholding to further reduce the bias towards the majority group. Several techniques which combine ensemble methods with sampling and cost-sensitive learning were presented in [28]. Liu et al. [52] proposed two algorithms, *EasyEnsemble* and *BalanceCascade*, that learn multiple classifiers by combining subsets of the majority group with the minority group, creating pseudo-balanced training sets for each individual classifier. *SMOTEBoost* [53], *DataBoost-IM* [54], and *JOUS-Boost* [55] all combine sampling with ensembles. Sun [56] introduced three cost-sensitive boosting methods, namely *AdaC1, AdaC2,* and *AdaC3*. These methods iteratively increase the impact of the minority group by introducing cost items into the *AdaBoost* algorithm's weight updates. Sun showed that the cost-sensitive boosted ensembles outperformed plain boosting methods in most cases.

Deep learning background

This section reviews the basic concepts of deep learning, including descriptions of the neural network architectures used throughout the surveyed works and the value of representation learning. We also touch on several important milestones that have contributed to the success of deep learning. Finally, the rise of *big data analytics* and its challenges are introduced along with a discussion on the role of deep learning in solving these challenges.

Introduction to deep learning

Deep learning is a sub-field of machine learning that uses *artificial neural networks* (ANNs) containing two or more hidden layers to approximate some function f^*, where f^* can be used to map input data to new representations or make predictions. The ANN, inspired by the biological neural network, is a set of interconnected neurons, or nodes, where connections are weighted and each neuron transforms its input into a single output by applying a non-linear activation function to the sum of its weighted inputs. In a feedforward network, input data propagates through the network in a forward pass, each hidden layer receiving its input from the previous layer's output, producing a final output that is dependent on the input data, the choice of activation function, and the weight parameters [1]. Gradient descent optimization is then used to adjust the network's weight parameters in order to minimize the loss function, i.e. the error between expected output and actual output.

The *multilayer perceptron* (MLP) is a fully-connected feedforward neural network containing at least one hidden layer. A shallow and deep MLP are illustrated in Fig. 1. The deep MLP is the simplest deep learning model in terms of implementation, but it quickly becomes very resource intensive as the number of weighted connections quickly increases.

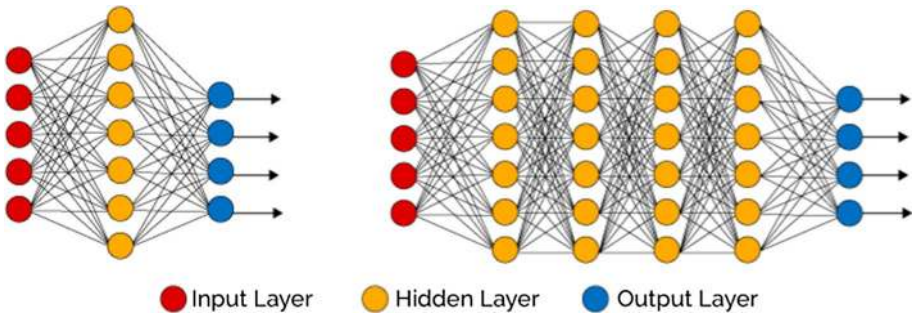

Fig. 1 Shallow MLP vs deep MLP [57]

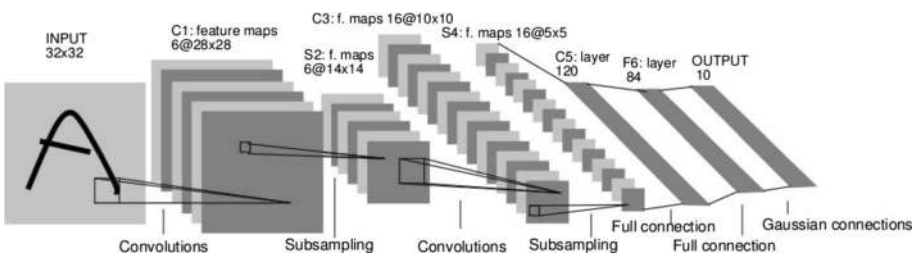

Fig. 2 LeNet-5 CNN architecture of 1998 [58]

The *convolutional neural network* (CNN) is a specialized feedforward neural network that was designed to process multi-dimensional data, e.g. images [58]. It was inspired by the brain's visual cortex and its origins date back to the *Neocognitron* presented by Fukushima in 1980 [59]. A CNN architecture is typically comprised of convolutional layers, pooling (subsampling) layers, and fully-connected layers. Figure 2 illustrates the LeNet-5 CNN architecture proposed by LeCun et al. [58] in 1998 for the purpose of character recognition. Unlike fully-connected layers, a single unit of a convolutional layer is only connected to a small *receptive field* of its input, where the weights of its connections define a *filter bank* [11]. The convolution operation is used to slide the filter bank across the input, producing activations at each receptive field that combine to form a *feature map* [60]. In other words, the same set of weights are used to detect a specific feature, e.g. a horizontal line, at each receptive field of the input, and the output feature map indicates the presence of this feature at each location. The concept of local connections and shared weights take advantage of the fact that input signals in close proximity of each other are usually highly correlated, and that input signals are often invariant to location. By combining multiple filter banks in a single convolutional layer, the layer can learn to detect multiple features in the input, and the resulting feature maps become the input of the next layer. Pooling layers are added after one or more convolutional layers in order to merge semantically similar features and reduce dimensionality [11]. After the convolutional and pooling layers, the multi-dimensional output is flattened and fed to fully-connected layers for classification. Similar to the MLP, output activations are fed from

one layer to the next in a forward pass, and the weights are updated through gradient descent.

The MLP and CNN are just two of many alternative DNN architectures that have been developed over the years. Recurrent neural networks (RNNs), autoencoders, and stochastic networks are explained thoroughly in [1, 60, 61]. They also present advanced optimization techniques that have been shown to improve training time and performance, e.g. regularization methods, parameter initialization, improved optimizers and activation functions, and normalization techniques.

Representation learning

The success of a conventional machine learning algorithm is highly dependent on the representation of the input data, making feature engineering a critical step in the machine learning workflow. This is very time consuming and for many complex problems, e.g. image recognition, it can be extremely difficult to determine which features will yield the best results. Deep learning offers a solution to this problem by building upon the concept of representation learning [11].

Representation learning is the process of using machine learning to map raw input data features into a new representation, i.e. a new feature space, for the purpose of improving detection and classification tasks. This mapping from raw input data to new representations is achieved through non-linear transformations of the input data. Composing multiple non-linear transformations creates hierarchical representations of the input data, increasing the level of abstraction through each transformation. This automatic generation of new features saves valuable time by removing the need for experts to manually hand engineer features, and improves overall performance in many complex problem domains, such as image and speech, where it is otherwise difficult to determine the best features. As data passes through the hidden layers of a DNN, it is transformed by each layer into a new representation. Given sufficient data, DNNs are able to learn high-level feature representations of inputs through the composition of multiple hidden layers. These learned representations amplify components of the input which are important for discrimination, while suppressing those that are unimportant [11]. Deep learning architectures achieve their power through this composition of increasingly complex abstract representations [60]. This approach to problem solving intuitively makes sense, as composing simple concepts into complex concepts is analogous to many real-world problem domains.

Deep learning milestones

The first DNNs date back to the 1960's, but they were largely abandoned in favor of traditional machine learning methods due to difficulties in training and inadequate performance [62]. In 1986, Rumelhart et al. [63] presented backpropagation, a method for efficiently updating neural network weight parameters by propagating the gradient of the loss function through multiple layers. It was believed by most, however, that gradient descent would be unable to escape poor local minima during optimization, preventing neural networks from converging to a global acceptable solution. Today, we believe this to be untrue, as theoretical results suggest that local minima are generally not an issue

and that systems nearly always reach solutions of similar quality [11]. Despite some early successes in the late 1980s [64] and 1990s [58], DNNs were mostly forsaken in practice and research due to these challenges.

In 2006, interests in deep learning were revived as research groups presented methods for sensibly initializing DNN weights with an unsupervised layer-wise pre-training procedure [65, 66]. These pre-trained *Deep Belief Networks* (DBNs) can then be efficiently fine-tuned through supervised learning. They proved to be very effective in image and speech tasks, and led to record breaking results on a speech recognition task in 2009 and the deployment of deep learning speech systems in Android mobile devices by 2012 [11].

In 2012, Krizhevsky et al. [17] submitted a deep CNN to the *Large Scale Visual Recognition Challenge* (LSVRC) [67] that nearly halved the top-5% error rate, reducing from the previous year's 26% down to just 16%. The work by Krizhevsky et al. included several crucial methods which have since become common practice in deep learning work. The CNN was implemented on multiple *graphics processing units* (GPUs). The drastic speed-up provided by parallel GPU computing allows for the training of deeper networks with larger data sets, and increased research productivity. A new non-saturating activation function, the *rectified linear unit* (ReLU), alleviated the vanishing gradient problem and allowed for faster training. *Dropout* was introduced as a regularization method to decrease over fitting in high capacity networks with many layers. Dropout simulates the ensembling of many models by randomly disabling neurons with a probability $P \in [0, 1]$ during each iteration, forcing the model to learn more robust features. Data augmentation, artificially enlarging the data set by applying transformations to data samples, was also applied as a regularization technique. This event marked a major turning point and sparked new interest in deep learning and computer vision.

This newfound interest in deep learning drove leading technological companies to increase research efforts, producing many advances in deep learning and pushing the state-of-the-art in deep learning to new levels. Deep learning frameworks which abstract tensor computation [12–15] and GPU compatibility libraries [16] have been made available to the community through open source software [68] and cloud services [69, 70]. Combined with an increasing amount of available data and public attention, deep learning is growing at a faster pace than ever before.

Deep learning with big data

Many organizations are being faced with the challenges of big data, as they are exploring large volumes of data to extract value and guide decisions [71]. Big data refers to data which exceeds the capabilities of standard data storage and data processing systems [72]. This forces practitioners to adopt new techniques for storing, manipulating, and analyzing data. The rise of big data can be attributed to improvements in hardware and software, increased internet and social media activity, and a growing abundance of sensor-enabled interconnected devices, i.e. the *internet of things* (IoT).

More specifically, big data can be characterized by the four Vs: volume, variety, velocity, and veracity [72, 73]. The large volumes of data being collected require highly scalable hardware and efficient analysis tools, often demanding distributed implementations. In addition to adding architecture and network overhead, distributed systems have been

shown to exacerbate the negative effects of class imbalanced data [74]. Advanced techniques for quickly processing incoming data streams and maintaining appropriate turnaround times are required to keep up with the rate at which data is being generated, i.e. data velocity. The variety of big data corresponds to the mostly unstructured, diverse, and inconsistent representations that arise as data is consumed from multiple sources over extended periods of time. This variety further increases the computational complexity of data preprocessing and machine learning. Finally, the veracity of big data, i.e. its accuracy and trustworthiness, must be regularly validated to ensure results do not become corrupted by invalid input. Some additional machine learning challenges that are magnified by big data include high-dimensionality, distributed infrastructures, real-time requirements, feature engineering, and data cleansing [75].

Najafabadi et al. [75] discuss the use of deep learning in solving big data challenges. The ability of DNNs to extract meaningful features from large sets of unlabeled data is particularly important, as this is commonly encountered in big data analytics. The automatic extraction of features from mostly unstructured and diverse data, e.g. image, text and audio data, is therefore extremely useful. With abstract features extracted from big data through deep learning methods, simple linear models can often be used to complete machine learning tasks more efficiently. Advanced semantic-based information storage and retrieval systems, e.g. semantic indexing and hashing [76, 77], are also made possible with these high-level features. In addition, deep learning has been used to tag incoming data streams, helping to group and organize fast-moving data [75]. In general, high-capacity DNNs are well suited for learning from the large volumes of data encountered in big data analytics.

As the presence of big data within organizations continues to increase, new methods will be required to keep up with the influx of data. Despite being relatively immature, deep learning methods are proving effective in solving many big data challenges. We believe that advances in deep learning, especially in learning from unsupervised data, will play a critical role in the future of big data analytics.

Deep learning methods for class imbalanced data

Anand et al. [78] explored the effects of class imbalance on the backpropagation algorithm in shallow neural networks in the 1990's. The authors show that in class imbalanced scenarios, the length of the minority class's gradient component is much smaller than the length of the majority class's gradient component. In other words, the majority class is essentially dominating the net gradient that is responsible for updating the model's weights. This reduces the error of the majority group very quickly during early iterations, but often increases the error of the minority group and causes the network to get stuck in a slow convergence mode.

This section analyzes a number of deep learning methods for addressing class imbalance, organized by data-level, algorithm-level, and hybrid methods. For each surveyed work, we summarize the implementation details and the characteristics of the data sets used to evaluate the method. We then discuss various strengths and weaknesses, considering topics such as class imbalance levels, interpretation of results, relative performance, difficulty of use, and generalization to other architectures and problem domains. Known limitations are highlighted and suggestions for future work are offered. For

consistency, class imbalance is presented as the maximum between-class ratio, ρ (Eq. 1), for all surveyed works.

Data-level methods

This section includes four papers that explore data-level methods for addressing class imbalance with DNNs. Hensman and Masko [79] first show that balancing the training data with ROS can improve the classification of imbalanced image data. Then RUS and augmentation methods are used by Lee et al. [20] to decrease class imbalance for the purpose of pre-training a deep CNN. Pouyanfar et al. [21] introduce a new dynamic sampling method that adjusts sampling rates according to class-wise performance. Finally, Buda et al. [23] compare RUS, ROS, and two-phase learning across multiple imbalanced image data sets.

Balancing training data with ROS

Hensman and Masko [79] explored the effects of class imbalance and ROS using deep CNNs. The *CIFAR-10* [80] benchmark data set, comprised of 10 classes with 6000 images per class, was used to generate 10 imbalanced data sets for testing. These 10 generated data sets contained varying class sizes, ranging between 6% and 15% of the total data set, producing a max imbalance ratio $\rho = 2.3$. In addition to varying the class size, the different distributions also varied the number of minority classes, where a minority class is any class smaller than the largest class. For example, a major 50–50 split (Dist. 3) reduced five of the classes to 6% of the data set size and increased five of the classes to 14%. As another example, a major singular over-representation (Dist. 5) increased the size of the airplane class to 14.5%, reducing the other nine classes slightly to 9.5%.

A variant of the AlexNet [17] CNN, which has proven to perform well on CIFAR-10, was used for all experiments by Hensman and Masko. The baseline performance was defined by training the CNN on all distributions with no data sampling. The ROS method being evaluated consisted of randomly duplicating samples from the minority classes until all classes in the training set had an equal number of samples.

Hensman and Masko presented their results as the percentage of correct answers per class, and included the mean score for all classes, denoted by Total. To ensure results were valid, a total of three runs were completed for each experiment and then averaged. Table 3 shows the results of the CNN without any data sampling. These results demonstrate the impact of class imbalance when training a CNN model. Most of the imbalanced distributions saw a loss in performance. Dist. 6 and Dist. 7, which contained very slight imbalance and no over-representation, performed just as well as the original balanced distribution. Some of the imbalanced distributions that contained over-represented classes, e.g. Dist. 5 and Dist. 9, yielded useless models that were completely biased towards the majority group.

Table 4 includes the results of training the CNN with balanced data that was generated through ROS. It shows that over-sampling performs significantly better than the baseline results from Table 3. Dist. 1 is excluded from Table 4 because it is already balanced, i.e. ROS is not applicable. In this experiment, ROS improved the classification results for all distributions. Dist. 5 and Dist. 9 saw the largest performance gains, increasing

Table 3 Imbalanced CIFAR-10 classification [79]

	Total	Airplane	Automobile	Bird	Cat	Deer	Dog	Frog	Horse	Ship	Truck
Dist. 1	0.73	0.78	0.84	0.62	0.57	0.70	0.62	0.80	0.76	0.84	0.80
Dist. 2	0.69	0.74	0.75	0.58	0.33	0.58	0.65	0.84	0.78	0.87	0.79
Dist. 3	0.66	0.71	0.75	0.59	0.30	0.52	0.61	0.79	0.77	0.85	0.73
Dist. 4	0.27	0.78	0.24	0.12	0.08	0.19	0.24	0.33	0.27	0.21	0.27
Dist. 5	0.10	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Dist. 6	0.73	0.74	0.86	0.65	0.53	0.71	0.63	0.81	0.76	0.83	0.79
Dist. 7	0.73	0.75	0.86	0.66	0.52	0.71	0.63	0.80	0.78	0.84	0.79
Dist. 8	0.66	0.63	0.75	0.55	0.35	0.51	0.58	0.82	0.74	0.84	0.80
Dist. 9	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00
Dist. 10	0.69	0.75	0.77	0.56	0.42	0.66	0.63	0.76	0.70	0.81	0.79
Dist. 11	0.69	0.74	0.82	0.58	0.44	0.59	0.64	0.80	0.69	0.83	0.81

Table 4 Imbalanced CIFAR-10 classification with ROS [79]

	Total	Airplane	Automobile	Bird	Cat	Deer	Dog	Frog	Horse	Ship	Truck
Dist. 2	0.72	0.77	0.80	0.57	0.51	0.68	0.64	0.81	0.78	0.84	0.82
Dist. 3	0.73	0.73	0.80	0.59	0.53	0.63	0.65	0.82	0.81	0.87	0.83
Dist. 4	0.73	0.76	0.82	0.60	0.54	0.68	0.63	0.81	0.78	0.85	0.83
Dist. 5	0.73	0.80	0.84	0.61	0.55	0.68	0.63	0.82	0.79	0.82	0.81
Dist. 6	0.73	0.75	0.86	0.65	0.51	0.65	0.66	0.81	0.78	0.85	0.80
Dist. 7	0.73	0.73	0.85	0.62	0.51	0.71	0.66	0.81	0.79	0.86	0.80
Dist. 8	0.73	0.78	0.84	0.62	0.56	0.66	0.64	0.81	0.77	0.83	0.80
Dist. 9	0.72	0.73	0.84	0.58	0.55	0.68	0.59	0.79	0.78	0.83	0.82
Dist. 10	0.73	0.78	0.85	0.63	0.50	0.67	0.63	0.82	0.75	0.87	0.80
Dist. 11	0.72	0.82	0.87	0.58	0.64	0.64	0.49	0.78	0.69	0.84	0.80

from baseline Total F1-scores of 0.10 up to 0.73 and 0.72, respectively. The ROS classification results for distributions Dist. 2–Dist. 11 are comparable to the results achieved by the baseline CNN on Dist. 1, suggesting that ROS has completely restored model performance.

The experiments by Hensman and Masko show that applying ROS to the level of class balance can be effective in addressing slight class imbalance in image data. It is also made clear by some of the results in Table 3 that small levels of imbalance are able to prevent a CNN from converging to an acceptable solution. We believe that the low imbalance levels tested ($\rho = 2.3$) is the biggest limitation of this experiment, as imbalance levels are typically much higher in practice. Besides exploring additional data sets and higher levels of imbalance, one area worth pursuing further is the total number of epochs completed during training on the imbalanced data. In these experiments, only 10 epochs over the training data were completed because the authors were more interested in comparing performance than they were in achieving high performance. Running additional epochs would help to rule out whether or not the poor performance was due to the slow convergence phenomenon described by Anand et al.

At the time of writing, May 2015, Hensman and Masko observed no existing research that examined the impact of class imbalance on deep learning with popular benchmark data sets. Our literature review also finds this to be true, confirming that addressing class imbalance with deep learning is still relatively immature and understudied.

Two-phase learning

Lee et al. [20] combined RUS with transfer learning to classify highly-imbalanced data sets of plankton images, WHOI-Plankton [81]. The data set contains 3.4 million images spread over 103 classes, with 90% of the images comprised of just five classes and the 5th largest class making up just 1.3% of the entire data set. Imbalance ratios of $\rho > 650$ are exhibited in the data set, with many classes making up less than 0.1% of the data set. The proposed method is the two-phase learning procedure, where a deep CNN is first pre-trained with thresholded data, and then fine-tuned using all data. The thresholded data sets for pre-training are constructed by randomly under-sampling large classes until they reach a threshold of N examples. The authors selected a threshold of $N = 5000$ through preliminary experiments, then down-sampled all large classes to N samples. The proposed model (G) was compared to six alternative methods (A–F), a combination of transfer learning and augmentation techniques, using unweighted average F1-scores to compare results.

(A) Full: CNN trained with original imbalanced data set.
(B) Noise: CNN trained with augmented data, where minority classes are duplicated through noise injection until all classes contain at least 1000 samples.
(C) Aug: CNN trained with augmented data, where minority classes are duplicated through rotation, scaling, translation, and flipping of images until all classes contain at least 1000 samples.
(D) Thresh: CNN trained with thresholded data, generated through random under-sampling until all classes have at most 5000 samples.
(E) Noise + full: CNN pre-trained with the noise-augmented data from (B), then fine-tuned using the full data set.
(F) Aug + full: CNN pre-trained with the transform-augmented data from (C), then fine-tuned using the full data set.
(G) Thresh + full: CNN pre-trained using the thresholded data set from (D), then fine-tuned using the full imbalanced data set.

Results from Lee et al.'s experiments are presented in Table 5, where the L5 and Rest columns are the unweighted average F1-scores for the largest five classes and the remaining minority classes, respectively. Comparing methods (B–D) shows that under-sampling (D) outperforms both noise injection (B) and augmentation (C) over-sampling methods on the given data set for all classes. Methods (B–D) saw moderate improvements on the Rest group when compared to the baseline (A), but the L5 group suffered sufficient loss in performance. Re-training the model with the full data set in (E–G) allowed the models to re-capture the distribution of the L5 group. The proposed model (G) achieved the highest F1-score (0.7791) on the L5 group, and

Table 5 Two-phase learning with WHOI-Plankton (Avg. F1-score) [20]

Classifier	All classes	L5	Rest
(A) Full	0.1773	0.7773	0.1548
(B) Noise	0.2465	0.5409	0.3599
(C) Aug	0.2726	0.5776	0.3700
(D) Thresh	0.3086	0.6510	0.4044
(E) Noise + full	0.3038	0.7531	0.2971
(F) Aug + full	0.3212	0.7668	0.3156
(G) Thresh + full	0.3339	0.7791	0.3262

greatly improved the F1-score of the Rest group from 0.1548 to 0.3262 with respect to the baseline.

The two-phase learning procedure presented by Lee et al. has proven effective in increasing the minority class performance while still preserving the majority class performance. Unlike plain RUS, which completely removes potentially useful information from the training set, the two-phase learning method only removes samples from the majority group during the pre-training phase. This allows the minority group to contribute more to the gradient during pre-training, and still allows the model to see all of the available data during the fine-tuning phase. The authors did not include details on the pre-training phase, such as the number of pre-training epochs or the criteria used to determine when pre-training was complete. These details should be considered in future works, as pre-training that results in high bias or high variance will certainly impact the final model's class-wise performance. Future work can also consider a hybrid approach, where the model is pre-trained with data that is generated through a combination of under-sampling majority classes and augmenting minority classes.

The previous year, Havaei et al. [82] used a similar two-phase learning procedure to manage class imbalance when performing brain tumor image segmentation. The brain tumor data contains minority classes that make up less than 1% of the total data set. Havaei et al. stated that the two-phase learning procedure was critical in dealing with the imbalanced distribution in their image data. The details of this paper are not included in this survey because the two-phase learning is just one small component of their domain-specific experiments.

Dynamic sampling

Pouyanfar et al. [21] used a dynamic sampling technique to perform classification of imbalanced image data with a deep CNN. The basic idea is to over-sample the low performing classes and under-sample the high performing classes, showing the model less of what it has already learned and more of what it does not understand yet. This is somewhat analogous to how humans learn, by moving on from easy tasks once learned and focusing attention on the more difficult tasks. The author's self-collected data set contains over 10,000 images captured from publicly available network cameras, including a total of 19 semantic concepts, e.g. intersection, forest, farm, sky, water, playground, and park. From the original data set, 70% is used for training models, 20% is used for validation, and 10% is set aside for testing. The authors report imbalance ratios in the data set as high as $\rho = 500$. Average F1-scores and weighted average F1-scores are used to compare the proposed model to a baseline CNN (A) and four alternative methods for handling class imbalance (B–E).

The system presented by Pouyanfar et al. includes three core components: real time data augmentation, transfer learning, and a novel dynamic sampling method. Real time data augmentation improves generalization by applying various transformations to select images in each training batch. Transfer learning is achieved by fine-tuning an Inception-V3 network [83] that was pre-trained using ImageNet [84] data. The dynamic sampling method is the main contribution relative to class imbalance.

$$\textit{Sample size}\ (F1_i, c_j) = \frac{1 - f1_{i,j}}{\sum_{c_k \in C}(1 - f1_{i,k})} \times N^* \tag{11}$$

$F1_i$ is a vector containing all individual class F1-scores after iteration i, and $f1_{i,j}$ denotes the F1-score for class j on iteration i, where F1-scores are calculated for each class in a one-versus-all manner. During the next iteration, classes with lower F1-scores are sampled at a higher rate, forcing the learner to focus more on examples previously misclassified. Eq. 11 is used to obtain the next iteration's sample size for a given class c_j, where N^* is the average class size. To prevent over-fitting of the minority group, a second model is trained through transfer learning without sampling. At time of inference, the output label is computed as a function of both models.

The proposed model is compared with the following alternative methods:

(A) Basic CNN: VGGNet [85] CNN trained on entire data set.
(B) Deep CNN features + SVM: Support vector machine (SVM) classifier trained with deep features generated by CNN.
(C) Transfer learning without augmentation: Fine-tuned Inception-V3 with no data augmentation.
(D) Transfer learning with augmentation: Fine-tuned Inception-V3 with data augmentation.
(E) Transfer learning with balanced augmentation: Fine-tuned Inception-V3 with data augmentation that enforces class balanced training batches with over-sampling and under-sampling.
(F) Proposed model: Dynamic sampling, data augmentation, and transfer learning on Inception-V3 network.

Average class-wise F1-scores are compared across all 19 concepts, showing that the basic CNN performs the worst in all cases. The basic CNN was unable to classify a single instance correctly for several concepts with very high imbalance ratios, including *Playground* and *Airport* with imbalance ratios of $\rho = 200$ and $\rho = 500$, respectively. Transfer learning methods (C–E) performed significantly better than the baseline CNN, increasing the weighted average F1-score from 0.630 to as high as 0.779. Results in Table 6 show that the proposed method (F) outperforms all other models tested on the given data set. Compared to transfer learning with basic augmentation (D), the dynamic sampling method (F) improved the weighted average F1-score from 0.779 to 0.794.

The dynamic sampling method's ability to self-adjust sampling rates is its most attractive feature. This allows the method to adapt to different problems containing varying levels of complexity and class imbalance, with little to no hyperparameter tuning. By removing samples that have already been captured by the network

Table 6 Dynamic sampling with network camera image data [21]

Classifier	Acc.	Avg. F1	WAvg. F1
(A) Basic CNN	0.649	0.254	0.630
(B) Deep CNN features + SVM	0.746	0.528	0.747
(C) TL + no aug.	0.765	0.432	0.755
(D) TL + basic aug.	0.792	0.502	0.779
(E) TL + balanced aug.	0.759	0.553	0.766
(F) Proposed model	0.802	0.599	0.794

parameters, gradient updates will be driven by the more difficult positive class samples. The dynamic sampling method outperforms a hybrid of over-sampling and under-sampling (E) according to F1-scores, but the details of the sampling method are not included in the description. We also do not know how dynamic sampling performs against plain RUS and ROS, as these methods were not tested. This should be examined closely in future works to determine if dynamic sampling can be used as a general replacement for RUS and ROS. One area of concern is the method's dependency on a validation set to calculate the class-wise performance metrics that are required to determine sampling rates. This will certainly be problematic in cases of class rarity, where very few positive samples exist, as setting aside data for validation may deprive the model of valuable training data. Methods for maximizing the total available training data should be included in future research. In addition, future research should extend the dynamic sampling method to non-CNN architectures and other domains.

ROS, RUS, and two-phase learning

Buda et al. [23] compare ROS, RUS, and two-phase learning using three multi-class image data sets and deep CNNs. MNIST [86], CIFAR-10, and ImageNet data sets are used to create distributions with varying levels of imbalance. Both MNIST and CIFAR-10 training sets contain 50,000 images spread evenly across 10 classes, i.e. 5000 images per class. Imbalanced distributions were created from MNIST and CIFAR-10 in the range of $\rho \in [10, 5000]$ and $\rho \in [2, 50]$, respectively. The ImageNet training data, containing 100 classes with a maximum of 1000 samples per class, was used to create imbalanced distributions in the range of $\rho \in [10, 100]$.

A different CNN architecture was empirically selected for each data set based off recent state-of-the-art results. For the MNIST and CIFAR-10 experiments, a version of the LeNet-5 [58] and the All-CNN [87] architectures were used for classification, respectively. Baseline results were established for each CNN architecture by performing classification on the data sets without any form of class imbalance technique, i.e. no data sampling or thresholding. Next, seven different methods for addressing class imbalance were integrated with the CNN architectures and tested. ROC AUC was extended to the multi-class problem by averaging the one-versus-all AUC for each class, and used to compare the methods. A portion of their results are presented in Fig. 3.

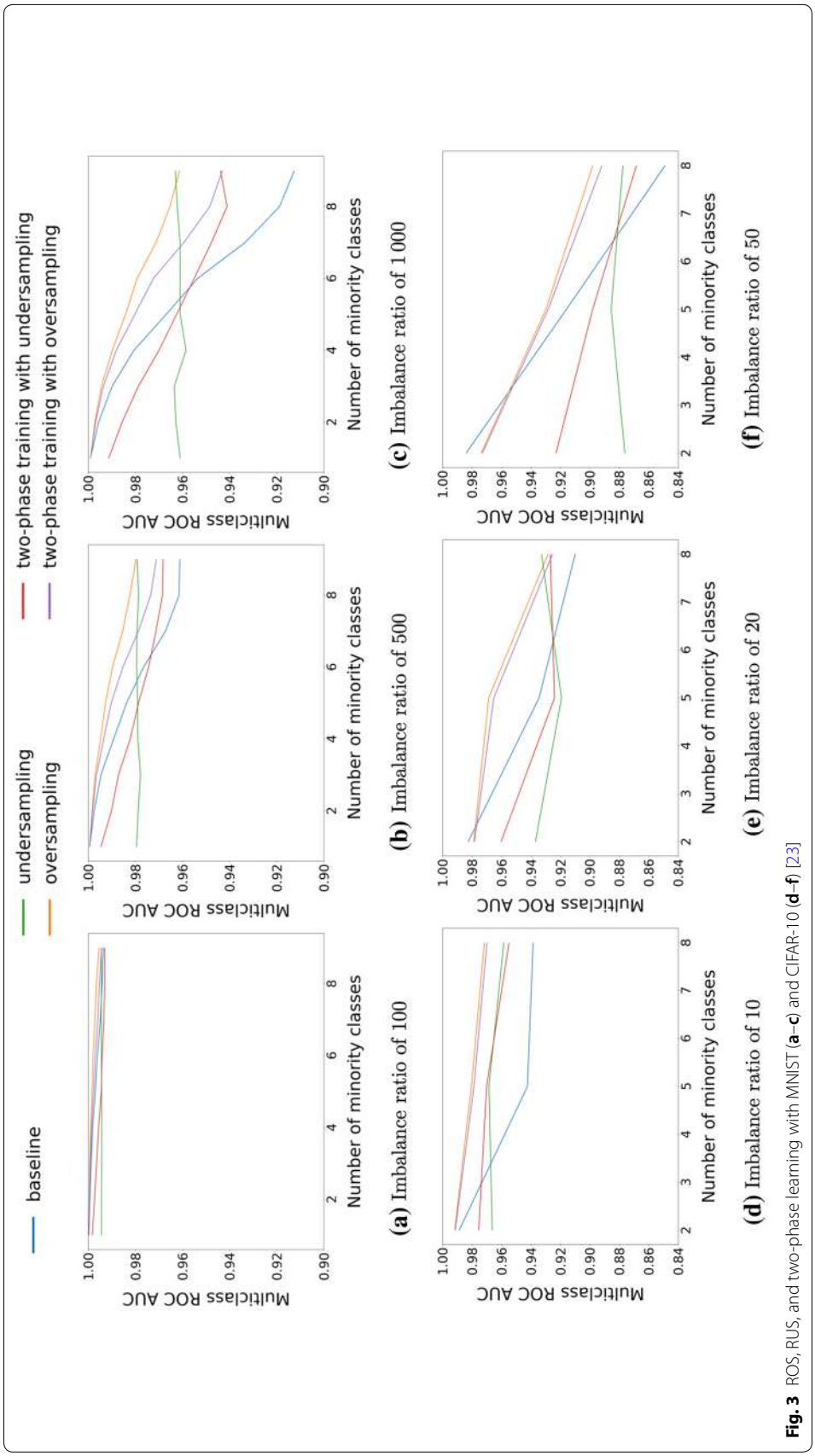

Fig. 3 ROS, RUS, and two-phase learning with MNIST (**a**–**c**) and CIFAR-10 (**d**–**f**) [23]

(A) ROS: All minority classes were over-sampled until class balance was achieved, where any class smaller than the largest class size is considered a minority class. In almost all experiments, over-sampling displayed the best performance, and never showed a decrease in performance when compared to the baseline.
(B) RUS: All majority classes were under-sampled until class balance was achieved, where any class larger than the smallest class size is considered a majority class. RUS performed poorly when compared to the baseline models, and never displayed a notable advantage to ROS. RUS was comparable to ROS only when the total number of minority classes was very high, i.e. 80–90%.
(C) Two-phase training with ROS: The model is first pre-trained on a balanced data set which is generated through ROS, and then fine-tuned using the complete data set. In general, this method performed worse than strict ROS (A).
(D) Two-phase training with RUS: Similar to (C), except the balanced data set used for pre-training is generated through RUS. Results show that this approach was less effective than RUS (B).
(E) Thresholding with prior class probabilities: The network's decision threshold is adjusted during the test phase based on the prior probability of each class, effectively shifting the output class probabilities. Thresholding showed improvements to overall accuracy, especially when combined with ROS.
(F) ROS and thresholding: The thresholding method (E) is applied after training the model with a balanced data set, where the balanced data set is generated through ROS. Thresholding combined with ROS performed better than the baseline thresholding (E) in most cases.
(G) RUS and thresholding: The thresholding method (E) is applied after training the model with a balanced data set, where the balanced data set is produced through RUS. Thresholding with RUS performed worse than (E) and (F) in all cases.

The work by Buda et al., which varies levels of class imbalance and problem complexity, is comprehensive, having trained almost 23,000 deep CNNs across three popular data sets. The impact of class imbalance was validated on a set of baseline CNNs, showing that classification performance is severely compromised as imbalance increases and that the impact of class imbalance seems to increase as problem complexity increases, e.g. CIFAR-10 versus MNIST. The authors conclude that ROS is the best overall method for addressing class imbalance, that RUS generally performs poorly, and that two-phase learning with ROS or RUS is not as effective as their plain ROS and RUS counterparts.

While it does provide a reasonable high-level view of each method's performance, the multi-class ROC AUC score provides no insight into the underlying class-wise performance trade-offs. It is not clear if there is high variance in the class-wise scores, or if one extremely low class-wise score is causing a large drop in the average AUC score. We believe that additional performance metrics, including class-wise scores, will better explain the effectiveness of each method in addressing class imbalance and help guide practitioners in model selection.

Buda et al. conclude that ROS should be performed until all class imbalance is eliminated. Despite their experimental results, we do not readily agree with this blanket statement, and argue that this is likely problem-dependent and requires further exploration.

The MNIST data set, both relatively low in complexity and small in size, was used to demonstrate that over-sampling until all classes are balanced performs best. We do not know how well over-sampling to this level will perform on more complex data sets, or in problems containing big data or class rarity. Furthermore, over-sampling to this level of class balance in a big data problem can be extremely resource intensive, drastically increasing training time by introducing large volumes of redundant data.

Summary of data-level methods

Two of the surveyed works [23, 79] have shown that eliminating class imbalance in the training data with ROS significantly improves classification results. Lee et al. have shown that pre-training DNNs with semi-balanced data generated through RUS or augmentation-based over-sampling improves minority group performance. In contrast to Lee et al., Buda et al. found that plain ROS and RUS generally perform better than two-phase learning. Unlike Lee et al., however, Buda et al. pre-trained their networks with data that was sampled until class balance was achieved. Since Lee et al. and Buda et al. used different imbalance levels for pre-training, and reported results with different performance metrics, it is difficult to understand the efficacy of two-phase learning. The dynamic sampling method outperformed baseline CNNs, but we do not know how it compares to ROS, RUS, or two-phase learning. Despite this limitation, the dynamic sampling method's ability to automatically adjust sampling rates throughout the training process is very appealing. Methods that can automatically adjust to varying levels of complexity and imbalance levels are favorable, as they reduce the number of tunable hyperparameters.

The experimental results suggest the use of ROS to eliminate class imbalance during the training of DNNs. This may be true for relatively small data sets, but we believe this will not hold true for problems containing big data or extreme class imbalance. Applying ROS until classes are balanced in very-large data sets, e.g. WHOI-Plankton data, will result in the duplication of large volumes of data and will drastically increase training times. RUS, on the other hand, reduces training time and may therefore be more practical in big data problems. We believe that RUS methods that remove redundant samples, reduce class noise, and strengthen class borders will prove helpful in these big data problems. Future work should explore these scenarios further.

All of the data-level methods presented were tested on class imbalanced image data with deep CNNs. In addition, differences in performance metrics and problem complexity make it difficult to compare methods directly. Future works should test these data-level methods on a variety of data types, imbalance levels, and DNN architectures. Multiple complementary performance metrics should be used to compare results, as this will better illustrate method trade-offs and guide future practitioners.

Algorithm-level methods

This section includes surveyed works that modify deep learning algorithms for the purpose of addressing class imbalance. These methods can be further divided into new loss functions, cost-sensitive learning, and threshold moving. Wang et al. [18] and Lin et al. [88] introduced new loss functions that allow the minority samples to contribute more to the loss. Wang et al. [89], Khan et al. [19], and Zhang et al. [90] experimented with cost-sensitive DNNs. The methods proposed by Khan et al. and Zhang et al. have the advantage of

learning cost matrices during training. The work by Buda et al. in the "ROS, RUS, and two-phase learning" section has also been included in this section, as they experimented with threshold adjusting. Zhang et al. [91] combine transfer learning, CNN feature extraction, and a cluster-based nearest neighbor rule to improve the classification of imbalanced image data. Finally, Ding et al. [92] experiment with very-deep CNNs to determine if increasing neural network depth improves convergence rates with imbalanced data.

Mean false error (MFE) loss

Wang et al. [18] found some success in modifying the loss function as they experimented with classifying imbalanced data with deep MLPs. A total of eight imbalanced binary data sets, including three image data sets and five text data sets, were generated from the CIFAR-100 [93] and 20 Newsgroup [94] collections. The data sets are all relatively small, with most training sets containing fewer than 2000 samples and the largest training set containing just 3500 samples. For each data set generated, imbalance ratios ranging from $\rho = 5$ to $\rho = 20$ were tested.

The authors first show that the mean squared error (MSE) loss function poorly captures the errors from the minority group in cases of high class imbalance, due to many negative samples dominating the loss function. They then propose two new loss functions that are more sensitive to the errors from the minority class, *mean false error* (MFE) and *mean squared false error* (MSFE). The proposed loss functions were derived by first splitting the MSE loss into two components, *mean false positive error* (FPE) and *mean false negative error* (FNE). The FPE (Eq. 12) and FNE (Eq. 13) values are then combined to define the total system loss, MFE (Eq. 14), as the sum of the mean error from each class.

Wang et al. introduce the MSFE loss (Eq. 15) as an improvement to the MFE loss, asserting that it better captures errors from the positive class. The MSFE can be expanded into the form $\frac{1}{2}((FPE + FNE)^2 + (FPE - FNE)^2)$, demonstrating how the optimization process minimizes the difference between FPE and FNE. The authors believe this improved version will better balance the error rates between the positive and negative classes.

$$FPE = \frac{1}{N}\sum_{i=1}^{N}\sum_{n}\frac{1}{2}(d_n^{(i)} - y_n^{(i)})^2 \tag{12}$$

$$FNE = \frac{1}{P}\sum_{i=1}^{P}\sum_{n}\frac{1}{2}(d_n^{(i)} - y_n^{(i)})^2 \tag{13}$$

$$MFE = FPE + FNE \tag{14}$$

$$MSFE = FPE^2 + FNE^2 \tag{15}$$

A deep MLP trained with the standard MSE loss is used as the baseline model. This same MLP architecture is then used to evaluate both the MFE and MSFE loss. Image classification results (Table 7) and text classification results (Table 8) show that the proposed models outperform the baseline in nearly all cases, with respect to F-measure and AUC scores.

Table 7 CIFAR-100 classification with MFE and MSFE [18]

Data set	Imbalance level (%)	F-measure			AUC		
		MSE	MFE	MSFE	MSE	MFE	MSFE
Household	20	0.3913	*0.4138*	*0.4271*	0.7142	*0.7397*	*0.7354*
	10	0.2778	*0.2797*	*0.3151*	0.7125	*0.7179*	*0.7193*
	5	0.1143	*0.1905*	*0.2353*	0.6714	*0.6950*	*0.6970*
Tree 1	20	0.5500	0.5500	0.5366	0.8100	*0.8140*	*0.8185*
	10	0.4211	0.4211	0.4211	0.7960	*0.7990*	*0.7990*
	5	0.1667	*0.2353*	*0.2353*	0.7920	*0.8000*	*0.8000*
Tree 2	20	0.4348	0.4255	0.4255	0.8480	0.8450	0.8440
	10	0.1818	*0.2609*	*0.2500*	0.8050	0.8050	*0.8060*
	5	0.0000	*0.1071*	*0.1481*	0.5480	*0.6520*	0.7000

Italic scores indicate MFE/MSFE loss outperforming MSE loss

Table 8 20 Newsgroup classification with MFE and MSFE[18]

Data set	Imbalance level (%)	F-measure			AUC		
		MSE	MFE	MSFE	MSE	MFE	MSFE
Doc. 1	20	0.2341	*0.2574*	*0.2549*	0.5948	*0.5995*	*0.5987*
	10	0.1781	*0.1854*	*0.1961*	0.5349	*0.5462*	*0.5469*
	5	0.1356	*0.1456*	*0.1456*	0.5336	*0.5436*	*0.5436*
Doc. 2	20	0.3408	0.3393	0.3393	0.6462	*0.6464*	*0.6464*
	10	0.2094	0.2000	0.2000	0.6310	*0.6319*	*0.6322*
	5	0.1256	0.1171	*0.1262*	0.6273	*0.6377*	*0.6431*
Doc. 3	20	0.2929	*0.2957*	*0.2957*	0.5862	*0.5870*	*0.5870*
	10	0.1596	*0.1627*	*0.1698*	0.5577	*0.5756*	*0.5865*
	5	0.0941	*0.1118*	*0.1084*	0.5314	*0.5399*	*0.5346*
Doc. 4	20	0.3723	*0.3843*	0.3668	0.6922	*0.7031*	*0.7054*
	10	0.1159	*0.2537*	*0.2574*	0.5623	*0.6802*	*0.6816*
	5	0.1287	*0.1720*	*0.1720*	0.6041	*0.6090*	*0.6090*
Doc. 5	20	0.3103	*0.3222*	*0.3222*	0.6011	0.5925	0.5925
	10	0.1829	0.1808	*0.1839*	0.5777	*0.5836*	*0.5837*
	5	0.0946	*0.1053*	*0.1053*	0.5682	*0.5730*	*0.5730*

Italic scores indicate MFE/MSFE loss outperforming MSE loss

Results show that the MFE and MSFE loss functions outperform MSE loss in almost all cases. Improvements over the baseline MSE loss are most apparent when class imbalance is greatest, i.e. imbalance levels of 5%. The MFE and MSFE performance gains are also more pronounced on the image data than on the text data. For example, the MSFE loss improved the classification of Household image data, increasing the F1-score from 0.1143 to 0.2353 when the class imbalance level was 5%.

Being relatively easy to implement and integrate into existing models is one of the biggest advantages of using custom loss functions for addressing class imbalance. Unlike data-level methods that increase the size of the training set, the loss function is less likely to increase training times. The loss functions should generalize to other domains with ease, but as seen in comparing the image and text performance results, performance gains will vary from problem to problem. Additional experiments should be performed

to validate MFE and MSFE effectiveness, as it is currently unclear how many rounds were conducted per experiment and the F1-score and AUC gains over the baseline are only minor improvements. On the image data experiments, for example, the average AUC gain over the baseline is just 0.025, with a median AUC gain over the baseline of only 0.008.

Focal loss

Lin et al. [88] proposed a model that effectively addresses the extreme class imbalance commonly encountered in object detection problems, where positive foreground samples are heavily outnumbered by negative background samples. Two-stage and one-stage detectors are well-known methods for solving such problems, where the two-stage detectors typically achieve higher accuracy at the cost of increased computation time. Lin et al. set out to determine whether a fast single-stage detector was capable of achieving state-of-the-art results on par with current two-stage detectors. Through analysis of various two-stage detectors (e.g. *R-CNN* [95] and its successors) and one-stage detectors (e.g. *SSD* [96] and *YOLO* [97]), class imbalance was identified as the primary obstacle preventing one-stage detectors from achieving state-of-the-art performance. The overwhelming number of easily classified negative background candidates create imbalance ratios commonly in the range of $\rho = 1000$, causing the negative class to account for the majority of the system's loss.

To combat these extreme imbalances, Lin et al. presented the *focal loss* (FL) (Eq. 16), which re-shapes the cross entropy (CE) loss in order to reduce the impact that easily classified samples have on the loss. This is achieved by multiplying the CE loss by a modulating factor, $\alpha_t(1-p_t)^\gamma$. Hyper parameter $\gamma \geq 0$ adjusts the rate at which easy examples are down weighted, and $\alpha_t \geq 0$ is a class-wise weight that is used to increase the importance of the minority class. Easily classified examples, where $p_t \to 1$, cause the modulating factor to approach 0 and reduce the sample's impact on the loss.

$$FL(p_t) = -\alpha_t(1-p_t)^\gamma \log(p_t) \tag{16}$$

Table 9 RetinaNet (focal loss) on COCO [88]

	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI	Inception-ResNet-v2	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	*52.1*
One-stage methods							
YOLOv2	DarkNet-19	21.6	44.0	19.2	5.0	22.4	35.5
SSD513	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet	ResNeXt-101-FPN	*40.8*	*61.1*	*44.1*	*24.1*	*44.2*	51.2

Italic scores indicate top AP performances

The proposed one-stage focal loss model, *RetinaNet*, is evaluated against several state-of-the-art one-stage and two-stage detectors. The RetinaNet model is composed of a backbone model and two subnetworks, where the backbone model is responsible for producing feature maps from the input image, and the two subnetworks then perform object classification and bounding box regression. The authors selected the *feature pyramid network* (FPN) [88], built on top of the ResNet architecture [98], as the backbone model and pre-trained it on ImageNet data. They found that the FPN CNN, which produces features at different scales, outperforms the plain ResNet in object detection. Two additional CNNs with separate parameters, the subnetworks, are then used to perform classification and bounding box regression. The proposed focal loss function is applied to the classification subnet, where the total loss is computed as the sum of the focal loss over all $\approx 100,000$ candidates. The COCO [99] data set was used to evaluate the proposed model against its competitors.

The first attempt to train RetinaNet using standard CE loss quickly fails and diverges due to the extreme imbalance. By initializing the last layer of the model such that the prior probability of detecting an object is $\pi = 0.01$, results improve significantly to an average precision (AP) of 30.2. Additional experiments are used to determine appropriate hyper parameters for the focal loss, selecting $\gamma = 2.0$ and $\alpha = 0.25$ for all remaining experiments.

Experiments by Lin et al. show that the RetinaNet, with the proposed focal loss, is able to outperform existing one-stage and two-stage object detectors. It outscores the runner-up one-stage detector (DSSD513 [100]) and the best two-stage detector (Faster R-CNN with TDM [101]) by 7.6-point and 4.0-point AP gains, respectively. When compared to several *online hard example mining* (OHEM) [102] methods, RetinaNet outscores the best method with an increase in AP from 32.8 to 36.0. Table 9 compares results between RetinaNet and seven state-of-the-art one-stage and two-stage detectors.

Lin et al. provide additional information to illustrate the effectiveness of the focal loss method. In one experiment, they use a trained model to compute the focal loss over $\approx 10^7$ negative images and $\approx 10^5$ positive images. By plotting the cumulative distribution function for positive and negative samples, they show that as γ increases, more and more weight is placed onto a small subset of negative samples, i.e. the hard negatives. In fact, with $\gamma = 2$, they show that most of the loss is derived from a very small fraction of samples, and that the focal loss is indeed reducing the impact of easily-classified negative samples on the loss. Run time statistics were included in the report to demonstrate that they were able to construct a fast one-stage detector capable of outperforming accurate two-stage detectors.

The new FL method lends itself to not just class imbalance problems, but also hard sample problems. It addresses the primary minority class gradient issue defined by Anand et al. by preventing the majority group from dominating the loss and allowing the minority group to contribute more to the weight updates. Similar to the MFE and MSFE loss functions, advantages include being relatively easy to integrate into existing models and having minimal impact on training time. We believe that the FL method's ability to down-weight easily-classified samples will allow it to generalize well to other domains. The authors compared FL directly to the CE loss, but we do not know how FL compares to other existing class imbalance methods. Future works should compare this

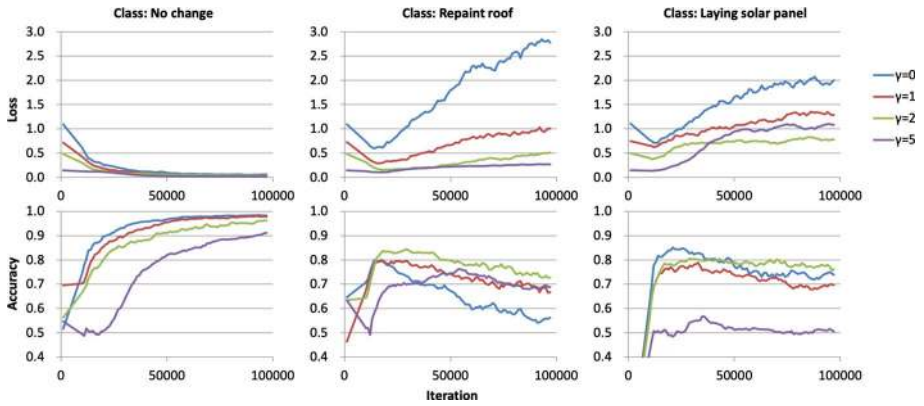

Fig. 4 Focal loss and building change detection [103]

loss function to alternative class imbalance methods across a variety of data sets and class imbalance levels.

Nemoto et al. [103] later used the focal loss in another image classification task, the automated detection of rare building changes, e.g. new construction. The airborne building images are annotated with the labels: no change, new construction, rebuilding, demolished, repaint roofs, and laying solar panel. The training data contains 203,358 images in total, where 200,000 comprise the negative class, i.e. no change. The repaint roofs and laying solar panel classes contain just 326 and 222 images, respectively, yielding class imbalance ratios as high as $\rho = 900$.

The experiments by Nemoto et al. utilize the VGG-16 [85] CNN architecture, where the baseline CNN uses the standard CE loss. Images were augmented by rotation and inversion, creating 20,000 samples per class. Class-wise accuracy on the validation set was used to compare the focal loss to the CE loss. The first experiment uses the CE loss function and shows that the accuracy on the repaint roof and laying solar panel classes begins to deteriorate after 25,000 iterations, suggesting over-fitting. In the second experiment, focal loss was evaluated using the same VGG-16 architecture and the same image augmentation procedure, generating 20,000 images per class. Unlike the first experiment, however, only three classes were selected for training and validation: no change, repaint roof, and laying solar panel images. The value of the focal loss's down-weighting parameter γ was varied in the range $\gamma \in [0, 5]$ to better understand its impact.

Figure 4 shows the training validation loss and accuracy of the focal loss method. Nemoto et al. conclude that focal loss improves problems related to class imbalance and over-fitting by adjusting the per-class learning speed. By comparing the loss plots, it is clear that as γ increases, the total loss decreases faster and the model is slower to overfit. It is not clear, however, if FL with $\gamma > 0$ is producing better classification results. First, the results from the first experiment cannot be compared to the results of the second experiment, because the total number of classes has been reduced from six to three, significantly reducing the complexity of the classification problem. Second, the results from the no change and laying solar panel classes in Fig. 4 show that the highest accuracy is achieved when $\gamma = 0$, where the laying solar panel class is the smallest class. The focal loss, with $\gamma = 0$, reduces to the standard cross entropy loss, suggesting that the CE

loss outperforms the focal loss on two of the three classes in this experiment. To better understand the effectiveness of FL, future work should include a baseline with consistent training data, several alternative methods for addressing class imbalance, and additional performance metrics.

Cost-sensitive deep neural network (CSDNN)

Wang et al. [89] employed a cost-sensitive deep neural network (CSDNN) method to detect hospital readmissions, a class imbalanced problem where a small percentage of patients are readmitted to a hospital shortly after their original visit. Two data sets were provided by the Barnes-Jewish Hospital, containing patient records spanning from 2007 to 2011. The first data set, general hospital wards (GHW), contains vital signs, clinical processes, demographics, real-time bedside monitoring, and other electronic data sources. Of the 2565 records in GHW, 406 patients were readmitted within 30 days and 538 patients were readmitted within 60 days, producing imbalance ratios of $\rho = 5.3$ and $\rho = 3.8$, respectively. The second data set, the operating room pilot data (ORP), contains vital signs, pre-operation data, laboratory tests, medical history, and procedure details. There are 700 records in the ORP data set, with 157 readmissions within 1 year and 124 readmissions within 30 days. The imbalance ratio of the ORP data is unclear; the authors merely state that the ORP data is less imbalanced than the GHW data ($\rho < 3.8$). A variety of performance metrics, including ROC, AUC, accuracy, recall, precision, positive predictive value (PPV), and negative predictive value (NPV) are used for evaluation. The PPV and NPV metrics are equivalent to positive and negative class precision scores. The proposed CSDNN method was found to outperform existing hospital readmission prediction systems, and has been deployed at the Barnes-Jewish Hospital.

Instead of representing categorical values with traditional one-hot encoding, Wang et al. use a categorical feature embedding approach to create more meaningful representations. In addition, they employed a CNN for automatic feature extraction from time series data, i.e. the patient vital signs data. The extracted features and categorical embeddings are concatenated, forming a final input feature vector that is fed to a DNN for classification. The DNN was composed of two hidden layers, with 128 and 64 neurons per layer. The CE loss function was modified to incorporate a pre-defined cost matrix, forcing the network to minimize misclassification cost. For the GHW data set, false negative errors were given a cost $2\times$ the cost of false positive errors. Similarly for the ORP data set, the false negative cost was set to $1.5\times$ the cost of false positive errors.

Wang et al.'s CSDNN method is compared to five baseline classifiers, including three decision tree methods, a SVM, and an ANN. Only one of the baseline methods addresses class imbalance, using under-sampling with a *Random Forest* (RF) classifier. The proposed method outperformed all of the baseline classifiers across all performance metrics except for NPV. We were able to observe the class-wise performance trade-off because multiple complementary performance metrics were reported. On the GHW data set, the CSDNN's AUC of 0.70 outscored the runner up's (ANN classifier) AUC of 0.62. Unfortunately, however, it is not possible to determine if the cost-sensitive loss function is the cause of the improved performance, as there are several other factors that may have contributed to improvements, e.g. categorical feature embedding and time-series

feature extraction. A baseline that includes the CNN feature extractor and the categorical embeddings is required to isolate the efficacy of the cost-sensitive loss function.

Incorporating the cost matrix into the CE loss is a minor implementation detail that should generalize well to other domains and architectures with minimal impact on training times. The process of identifying an ideal cost matrix is likely the biggest limitation. In traditional machine learning problems with relatively small data sets, models can be validated across a range of costs and the best cost matrix can be selected for the final model. When working with DNNs and large data sets, however, this process of searching for the best cost parameters can be very time consuming and even impractical. Including results from a range of cost matrices in future work will help to demonstrate the class-wise performance trade-offs that occur as costs vary.

Learning cost matrices with cost-sensitive CNN (CoSen)

Khan et al. [19] introduced an effective cost-sensitive deep learning procedure which jointly learns network weight parameters and class misclassification costs during training. The proposed method, *CoSen CNN*, is evaluated against six multi-class data sets with varying levels of imbalance: MNIST, CIFAR-100, Caltech-101 [104], MIT-67 [105], DIL [106], and MLC [107]. Class imbalance ratios of $\rho = 10$ are tested for the MNIST, CIFAR-100, Caltech-101, and MIT-67 data sets. The DIL and MLC data sets have imbalance ratios of $\rho = 13$ and $\rho = 76$, respectively. The VGG-16, pre-trained on ImageNet data, is used as a feature extractor and the baseline CNN throughout the experiments.

The cost matrix that is learned by the CoSen CNN is used to modify the output of the VGG-16 CNN's last layer, giving higher importance to samples with higher cost. The group presents three modified loss functions, incorporating the learned cost parameters into MSE loss, SVM hinge loss, and CE loss. The training process learns network weight parameters and misclassification cost parameters by keeping one fixed at a time, and minimizing cost with respect to the other during training. The cost matrix update is dependent on the current classification errors, the overall classification error, and the *class-to-class separability* (C2C). The C2C separability measures relationships between within-class sample distances and the size of class-separating boundaries.

Table 10 Cost-sensitive CoSen CNN results (accuracy) [19]

Dataset	Imbalance protocol	SMOTE (%)	RUS (%)	SMOTE RSB (%)	CoSen SVM (%)	CoSen RF (%)	SOSR CNN (%)	Baseline CNN (%)	CoSen CNN (%)
MNIST	10% of odd classes	94.5	92.1	96.0	96.8	96.3	97.8	97.6	*98.6*
CIFAR-100	10% of odd classes	32.2	28.8	37.5	39.9	39.0	55.8	55.0	*60.1*
Caltech-101	10% of odd classes	67.7	61.4	68.2	70.1	68.7	77.4	77.4	*83.2*
MIT-67	10% of odd classes	33.9	28.4	34.0	35.5	35.2	49.8	50.4	*56.9*
DIL	Standard split	50.3	46.7	52.6	55.3	54.7	68.9	69.5	*72.6*
MLC	Standard split	38.9	31.4	43.0	47.7	46.5	65.7	66.1	*68.6*

Italic scores indicate the top performance for each data set

The CoSen CNN is evaluated against the baseline CNN, multiple sampling methods, and multiple cost-sensitive methods. A two-layered neural network was used for the sampling classification methods, and a SVM and RF were used for the cost-sensitive methods. The SVM and RF baseline classifiers used the features that were extracted by the pre-trained VGG-16 CNN as input. The SOSR CNN is a cost-sensitive deep learning method that incorporates a fixed cost matrix into the loss function [108].

Overall accuracy was used to show that the proposed CNN outperforms all seven alternative techniques across all data sets. Table 10 shows that the CoSen CNN performed exceptionally well, outperforming the runner-up classifier by more than 5% on CIFAR-100, Caltech-101, and MIT-67. The second best classifier listed in the experimental results is between the SOSR CNN and the baseline CNN. In all cases SMOTE outperformed RUS, and hybrid sampling method SMOTE-RSB [109] outperformed SMOTE. Unfortunately, with accuracy being the only performance metric reported between all seven class imbalance methods, and accuracy being unreliable in cases of class imbalance, these results may be misleading.

F1 and G-Mean scores were reported to show that the proposed CoSen CNN outperforms the baseline CNN on all data sets, e.g. increasing the F1-score from 0.389 to 0.416 on the Caltech-101 data set. Khan et al. also included sample network training times, showing that the added cost parameter training increases each training epoch by several seconds but has little to no impact on the inference step. In another experiment, the authors defined three fixed cost matrices using class representation, data separability, and classification errors to derive the costs, and showed that the dynamic cost matrix method outperforms all three cases.

It is interesting that the baseline CNN, with no class imbalance modifications, is a close runner-up to the CoSen CNN, outperforming the sampling methods, SVM, and RF classifiers in all cases. This does not imply that a deep CNN with no class imbalance modifications is better equipped to address class imbalance than traditional sampling methods. Rather, this demonstrates the power of re-using strong feature extractors that have been trained on large volumes of data, e.g. over 1.3 million images in this experiment.

As mentioned previously, one of the difficulties of cost-sensitive learning is choosing an appropriate cost matrix, often requiring a domain expert or a grid search procedure. The CoSen method proposed by Khan et al. removes this requirement, and offers more of an end-to-end deep learning framework capable of learning from class imbalanced data. The authors report results on a number of experiments across a variety of data sets, and the joint learning of network parameters and cost parameters appears to be an excellent candidate for learning from class imbalanced data. Future experiments should explore this cost-sensitive method's ability to learn from problems containing alternative data types, big data, and class rarity.

Cost-sensitive DBN with differential evolution (CSDBN-DE)

Zhang et al. [90] set out to automatically learn costs for the purpose of cost-sensitive deep learning. More specifically, they use a differential evolutionary algorithm [110] to improve the cost matrix each training iteration, and incorporate these learned costs into a DBN. The proposed *cost-sensitive deep belief network with differential evolution*

(CSDBN-DE) is evaluated against 42 data sets from the *Knowledge Extraction based on Evolutionary Learning* (KEEL) [111] repository. The training data sets are structured data sets, ranging in size from 171 samples to 3339 samples and containing between 6 and 13 attributes each. Imbalance ratios range from $\rho = 9.28$ up to $\rho = 128.21$.

The DBN pre-training phase follows the original layer-wise training method first proposed by Hinton et al. [65]. The cost matrix is then incorporated into the output layer's softmax probabilities during the fine-tuning phase. Cost matrices are first randomly initialized, then training set evaluation scores are used to select a new cost matrix for the next population. Mutation and cross-over operations are applied to evolve and generate the next population of cost matrices. Once training is complete, the best cost matrix is selected and applied to the output layer of the DBN, forming the final version of the cost-sensitive DBN to be used for inference.

Accuracy and error rate are used to compare the proposed method to an *extreme learning machine* (ELM) [112]. The ELM network is a type of single-hidden layer feed-forward network that does not use backpropagation and is known to train thousands of times faster than neural networks. Experiments are repeated 20 times and results are averaged. The proposed CSDBN-DE model outperformed the ELM network on 28 out of 42 data sets, i.e. 66% of the experiments.

Similar to the CoSen CNN [19], the biggest advantage of the CSDBN-DE is its ability to learn an effective cost matrix throughout training, as this is often a difficult and time-consuming process. Unfortunately, it is not very clear how well the CSDBN-DE handles class imbalanced data, because it is compared to a single baseline built on a completely different architecture. Also, the accuracy performance metric provides no real insight into the network's true performance when class imbalance is present. The concept of incorporating an evolutionary algorithm to iteratively update a cost matrix shows promise, but more robust experiments are required to validate its ability to classify imbalanced data.

Output thresholding

In addition to the data sampling methods discussed in the "ROS, RUS, and two-phase learning" section, Buda et al. [23] experimented with adjusting CNN output thresholds to improve overall performance. They used the MNIST and CIFAR-10 data sets with varying levels of class imbalance ratios in the range of $\rho \in [1, 5000]$ and $\rho \in [1, 50]$, respectively. Accuracy scores were used to compare thresholding with the baseline CNN, ROS, and RUS methods as they were described in the "ROS, RUS, and two-phase learning" section.

The authors applied thresholding by dividing the network outputs for each class by its estimated prior probability, effectively reducing the likelihood of misclassifying examples from the minority group. They also considered hybrid methods by combining threshold moving with RUS and ROS. The thresholding method outperformed the baseline CNN for all levels of class imbalance on the MNIST and CIFAR-10 data sets. Thresholding outperformed ROS and RUS for all levels of imbalance on the MNIST data, but on the CIFAR-10 data there were several instances were ROS outperformed

thresholding. Thresholding combined with ROS performed especially well, outperforming all other methods in nearly all cases.

As discussed in the "ROS, RUS, and two-phase learning" section, Buda et al. explored a variety of deep learning methods for addressing a wide range of class imbalance levels. They have shown that overall accuracy can be improved with threshold moving, and that it can be implemented relatively easily with prior class probabilities. Unfortunately, the accuracy score does not explain individual class-wise performances and trade-offs. Since thresholding is only applied during inference, it does not affect training times. This also means that thresholding does not impact weight tuning and therefore does not improve a model's ability to discriminate between classes. Regardless, it is still an appropriate method for reducing majority class bias that can be quickly implemented on top of an already trained network to improve classification results.

Category centers

Zhang et al. [91] experimented with deep representation learning of class imbalanced image data from the CIFAR-10 and CIFAR-100 data sets. They present a method for addressing class imbalance, *category centers* (CC), that combines transfer learning, deep CNN feature extraction, and a nearest neighbor discriminator. Three class imbalanced distributions are created from each original data set through random under-sampling, i.e. Dist. A, Dist. B, and Dist. C. In Dist. A and Dist. B, half of the classes are reduced in size, creating imbalance levels of $\rho = 10$ and $\rho = 20$, respectively. In Dist. C, class reduction levels increase linearly across all classes with a max imbalance of $\rho = 20$. For example, Dist. C for the CIFAR-100 data set contains 25 images in each of the first 10 classes, 75 images in each of the next 10 classes, then 125 images in each of the next 10 classes, etc. The proposed model is compared to both a baseline CNN and an over-sampling method. The mean precision performance metric is used to compare results.

Zhang et al. observe that similar images of the same class tend to cluster well in CNN deep feature space. They discuss the decision boundary that is created by the final layer of the CNN, the classification layer responsible for separating these deep feature clusters, stating that there is a greater chance for large errors in boundary placement when class imbalance is present. To avoid this boundary error in scenarios of class imbalance, they propose using high-level features extracted by the CNN to calculate each class's centroid in deep feature space. These category centers in deep feature space can then be used to classify new images, by assigning new images to their nearest deep feature

Table 11 Category centers with CIFAR-10 data (AP) [91]

Classifier	Dist A ($\rho = 10$)	Dist B ($\rho = 20$)	Dist C ($\rho \in [1, 20]$)
(A) Baseline CNN	0.779	0.747	0.857
(B) CC last conv. layer	0.824	0.775	0.859
(C) CC last FC layer	0.826	0.772	0.865
(D) Baseline CNN + over-sampling	0.787	0.770	0.850
(E) CC last conv. layer + over-sampling	0.831	0.792	0.861
(F) CC last FC layer + over-sampling	0.830	0.796	0.862

category center. The VGG-16 network, pre-trained on ImageNet data, is used throughout the experiments as the baseline CNN. The proposed method fine-tunes the CNN on the imbalanced distribution, maps all images to their corresponding deep feature representations, then calculates the centroid for each class. At test time the trained CNN is used to extract features from new images, and each new image is assigned to the class of its nearest category center in feature space, defined by the Euclidean distance between the features. Zhang et al. claim that the category center is significantly more stable than the boundary generated by a CNN's classification layer, but no evidence was provided to support this.

A portion of Zhang et al.'s results are displayed in Table 11. The authors tried using two different CNN layers for feature extraction, the last convolutional layer (B) and the last fully connected layer (C), to determine if one set of features performed better. Finally, over-sampling was applied to the baseline and the category centers methods to determine the impact of data sampling on CC (D–F). These results show that the CC methods (B, C) outperform the baseline CNN (A) on mean precision for all three imbalance scenarios. Results also show that the addition of over-sampling (D–F) led to improved results on the CIFAR-10 distributions, with the exception of Dist. C. For example, the CC method with over-sampling (E) increased mean precision on Dist. A from a baseline (A) of 0.779 to 0.830. CIFAR-100 results were similar, in the sense that the proposed method (B, C) outperformed the baseline (A) for all three distributions. Unlike the CIFAR-10 data, interestingly, over-sampling did not improve the results of the proposed method when classifying the CIFAR-100 distributions, but it did improve the baseline CNN.

We believe that the biggest limitation to the CC method is that it is highly dependent on the DNN's ability to generate discriminative features that cluster well. If a large volume of class balanced, labelled training data is not available to pre-train the DNN, deep feature boundaries may not be strong enough to apply the CC method. The specifics of the over-sampling method used in methods (D–F) are unknown, so we do not know if over-sampling was applied to a level of class balance or some other ratio. In addition, we do not know if these results are the average of several rounds of experiments, or are just results from a single instance. With no balanced distribution results, an unclear over-sampling method, and only one performance metric, it is difficult to understand how well the proposed CC method does in learning from class imbalance.

Very-deep neural networks

Ding et al. [92] experimented with very-deep CNN architectures, e.g. 50 layers, to determine if deeper networks perform better on imbalanced data. The authors observe in literature that the error surfaces of deeper networks have better qualities for training convergence than smaller sized networks [113, 114]. They claim that larger networks contain more local minimum with good performance, making acceptable solutions easier to locate with gradient descent. The networks are tested on the problem of Facial Action Units (FAUs) recognition, i.e. the detection of basic facial actions as defined by the Facial Action Coding System [115].

The EmotioNet Challenge Track 1 data set [116] is used to compare methods. From the original data set, with over one million images, 450,000 images were randomly selected

Table 12 Comparing very-deep CNNs on FAU recognition (F1-score) [92]

FAU class	Imbalance level	(A) Handcraft	(B) Plain34	(C) Res10	(D) Res18	(E) Res34	(F) Res50
AU1	2.5%	0.05	0.43	0.46	0.44	0.44	0.46
AU2	1.5%	0.00	0.42	0.44	0.45	0.45	0.46
AU4	1.1%	0.00	0.44	0.42	0.41	0.40	0.42
AU5	0.9%	0.00	0.42	0.41	0.42	0.40	0.46
AU6	3.6%	0.41	0.52	0.51	0.55	0.52	0.51
AU9	0.6%	0.00	0.41	0.36	0.38	0.40	0.44
AU12	45.7%	0.89	0.89	0.90	0.89	0.90	0.90
AU17	0.4%	0.00	0.19	0.20	0.26	0.27	0.24
AU20	0.01%	0.00	0.00	0.00	0.00	0.00	0.00
AU25	29.9%	0.79	0.85	0.86	0.86	0.86	0.85
AU26	25.6%	0.64	0.73	0.74	0.73	0.74	0.73
Average	–	0.29	0.48	0.48	0.49	0.49	0.50

for training and 40,000 images were randomly selected for validation. The images in this data set contain 11 possible FAUs. Since an image can be positive for more than one FAU, the authors treated the classification problem as a set of 11 binary problems. Several FAUs are present in less than 1% of the data set, e.g. nose wrinkler, chin raiser, and upper lid raiser. The lip stretcher FAU is only present in 0.01% of the data, creating a max imbalance ratio to $\rho = 10,000$. Six deep CNN architectures are compared, all of which include 3×3 convolution filter layers and are trained for 100 epochs:

(A) Handcraft: 6-layer CNN
(B) Plain34: 34-layer CNN
(C) Res10: 10-layer ResNet CNN
(D) Res18: 18-layer ResNet CNN
(E) Res34: 34-layer ResNet CNN
(F) Res50: 50-layer ResNet CNN.

F1-scores from the first experiment are presented in Table 12. The shallow network (A) was unable to capture classes containing high imbalance, e.g. AU2–AU5, during the 100 training epochs. The non-residual 34-layer CNN (B) saw a large boost in performance compared to the shallow network (A), with average F1-score increasing from 0.29 to 0.48. It can also be observed that the 10-layer ResNet (C) achieved an equal F1-score of 0.48. The FAU with the largest imbalance, i.e. lip stretcher (AU20), received an F1-score of 0.0 in all experiments. There is no noticeable difference in F1-scores between the networks with 10 layers or more (B–F). The 34-layer ResNet (E) won first place at the first track EmotioNet Challenge in 2017 [116].

In a second experiment, Ding et al. compare the convergence rate of a very-deep 18-layer CNN to a shallower 6-layer CNN across 100 epochs. The experiment shows that the very-deep 18-layer CNN converges faster than the shallower network, as the error decreases at a faster rate. The authors suggest that the shallower network will continue to converge, provided additional epochs.

The experiments by Ding et al. show that additional hidden layers can increase the convergence rate on facial action recognition. Training very-deep neural networks

comes at a cost, however, as it increases the total number of matrix operations and the memory footprint. Additional experiments are required to determine if this increased convergence rate is observed with alternative deep learning architectures and data sets. With the convergence rate in question, other methods that have been shown to impact convergence rates should be included in future studies, e.g. alternative optimization methods, dynamic learning rates, and weight initialization.

Summary of algorithm-level methods

This section included eight algorithm-level methods for addressing class imbalance with DNNs. The MFE and MSFE loss functions presented by Wang et al. outperform the standard MSE loss on several image and text data sets. Lin et al.'s focal loss, which down-weights easy-to-classify samples, was used to outperform several state-of-the-art one-stage and two-stage object detectors on the COCO data set. Nemoto et al. also explored focal loss for the purpose of detecting rare building changes, but the results were somewhat contradictory. A cost-sensitive method that incorporates pre-defined misclassification costs into the CE loss function was used by Wang et al. to predict hospital readmissions. Khan et al. proposed a cost-sensitive deep CNN (CoSen) that jointly learns network parameters and cost matrix parameters during training. Overall accuracy was used to show that the CoSen CNN outperforms a baseline CNN, multiple sampling methods, and multiple cost-sensitive methods consistently across six image data sets. Similarly, Zhang et al. combined a DBN with an evolutionary algorithm that searches for optimal misclassification costs, but results were again reported with the accuracy metric and are difficult to interpret. Buda et al. demonstrated how prior class probabilities can be used to adjust DNN output thresholds to improve overall accuracy in classifying image data. Zhang et al. presented category centers, a method that addresses class imbalance by combining transfer learning, deep CNN feature extraction, and a nearest deep feature cluster discrimination rule. Finally, Ding et al. experimented with very-deep neural networks ($>$ 10 layers) and showed that deeper networks may converge faster due to changes in the error surfaces that allow for faster optimization.

Unlike data sampling methods, the algorithm-level methods presented do not alter the training data and do not require any pre-processing steps. Compared to ROS, which was recommended by multiple authors in the "Data-level methods" section, algorithm-level methods are less likely to impact training times. This suggests that algorithm-level methods may be better equipped for big data problems. With the exception of defining misclassification costs, the algorithm-level methods require little to no tuning. Fortunately, two methods were presented for automatically learning cost parameters. Methods which are able to adapt to different problems with minimal tuning are preferred, as they can be quickly applied to new problems and do not require specific domain knowledge. The focal loss function and CoSen CNN demonstrate this flexibility, and we believe they will generalize well to many complex problem domains.

In general, there is a lack of research that appropriately compares deep learning algorithm-level methods to alternative class imbalance methods. This is due to poor choices in baseline models, insufficient performance metrics, and domain-specific experiments that fail to isolate the proposed class imbalance method. Similar to the surveyed deep

learning data-level methods, most of the methods in this section were evaluated on image data with deep CNNs. We are most interested in understanding how deep learning methods for addressing class imbalance compare to each other and to traditional machine learning techniques for class imbalance. We believe that filling these research gaps and properly evaluating these methods will have a great impact on future deep learning applications.

Hybrid-methods

The deep learning techniques for addressing class imbalance in this section combine algorithm-level and data-level methods. Huang et al. [22] use a novel loss function and sampling method to generate more discriminative representations in their *Large Margin Local Embedding* (LMLE) method. Ando and Huang [117] presented the first deep feature over-sampling method, *Deep Over Sampling* (DOS). Finally, class imbalance in large-scale image classification is addressed by Dong et al. [118] with a novel loss function and hard sample mining.

Large Margin Local Embedding (LMLE)

Huang et al. [22] proposed the LMLE method for learning more discriminative deep representations of imbalanced image data. The method is motivated by the observation that minority groups are sparse and typically contain high variability, allowing the local neighborhood of these minority samples to be easily invaded by samples of another class. By combining a new informed *quintuplet sampling* method with a new *triple-header hinge loss* function, deep feature representations that preserve same class locality and increase inter-class discrimination are learned from imbalanced image data. These deep feature representations, which form well-defined clusters, are then used to label new samples with a fast cluster-wise K-NN classification method. The proposed LMLE method is shown to achieve state-of-the-art results on the CelebA [119] data set, which contains high imbalance levels up to $\rho = 49$.

The quintuplet sampling method selects an anchor and four additional samples based on inter-class and intra-class cluster distances. During training, each mini-batch selects an equal number of quintuplets from both the minority and majority classes. The five samples obtained by the quintuplet sampling method are fed to five identical CNNs, and their outputs are aggregated into a single result. The triple-header hinge loss is then used to compute the error and update the network parameters accordingly. This regularized loss function constrains the deep feature representation such that clusters collapse into small neighborhoods with appropriate margins between inter-class and intra-class clusters.

The CelebA data set contains facial images annotated with 40 attributes, with imbalance levels as high as $\rho = 49$ (Bald vs not Bald). A total of 160,000 images are used to train a CNN, and the learned deep representations are then fed to a modified K-NN classifier. The Balanced Accuracy (Eq. 9) metric is calculated for each facial attribute. The LMLE-kNN model is compared to three state-of-the-art models in the facial attribute recognition domain: Triplet-kNN [120], PANDA [121], and ANet [122]. Results in Table 13 show that the LMLE-kNN performs as good as, or better than, alternative methods for all facial attributes. Performance gains over alternative methods increased

Table 13 LMLE CelebA facial attribute recognition (Balanced Accuracy) [22]

	Attractive	Mouth open	Smiling	Wear lipstick	High cheekbones	Male	Heavy makeup	Wavy hair	Oval face	Pointy nose	Arched eyebrows	Black hair	Big lips	Big nose	Young	Straight hair	Brown hair	Bags under eyes	Wear earings	No beard	Bangs
Imbalance level	1	2	2	3	5	8	11	18	22	22	23	26	26	27	28	29	30	30	31	33	35
Triplet-kNN	83	92	92	91	86	91	88	77	61	61	73	82	55	68	75	63	76	63	69	82	81
PANDA	85	93	98	97	89	99	95	78	66	67	77	84	56	72	78	66	85	67	77	87	92
ANet	87	96	97	95	89	99	96	81	67	69	76	90	57	78	84	69	83	70	83	93	90
LMLE-kNN	88	96	99	99	92	99	98	83	68	72	79	92	60	80	87	73	87	73	83	96	98

	Blond hair	Bushy eyebrows	Wear necklace	Narrow eyes	5 o'clock shadow	Receding hairline	Wear necktie	Eyeglasses	Rosy cheeks	Goatee	Chubby	Sideburns	Blurry	Wear hat	Double chin	Pale skin	Gray hair	Mustache	Bald	Average
Imbalance level	35	36	38	38	39	42	43	44	44	44	44	44	45	45	45	46	46	46	48	
Triplet-kNN	81	68	50	47	66	60	73	82	64	73	64	71	43	84	60	63	72	57	75	72
PANDA	91	74	51	51	76	67	85	88	68	84	65	81	50	90	64	69	79	63	74	77
ANet	90	82	59	57	81	70	79	95	76	86	70	79	56	90	68	77	85	61	73	80
LMLE-kNN	99	82	59	59	82	76	90	98	78	95	79	88	59	99	74	80	91	73	90	84

as levels of imbalance increased, e.g. LMLE increased accuracy on the Bald attribute from 75 to 90 when compared to its runner-up. The authors achieved similar results when comparing the LMLE-kNN to 14 state-of-the-art models on the BSDS500 [123] edge detection data set.

Huang et al. were one of the first to study deep representation learning with class imbalanced data. The proposed LMLE method combines several powerful concepts, achieves state-of-the-art results on a popular image data benchmark, and shows potential for future works in other domains. The high-performance results do come with a cost, however, as the system is both complex and computationally expensive. The data must be pre-clustered before quintuplet sampling can take place, which may be difficult if a suitable feature extractor is not available for the given data set. The authors do suggest that the initial clustering is not important, because the learning procedure can be used to re-cluster the data throughout training. Furthermore, each forward pass through the system requires computation from five CNNs, one for each of the quintuplets sampled. We are in agreement with the statement made by Ando and Huang [117], that adapting LMLE to new problems will require many task and model specific configurations. This will likely deter many practitioners from using this method when trying to solve class imbalanced problems, as most will resort to simpler and faster methods. Results provided by Dong et al. in the "Class rectification loss (CRL) and hard sample mining" section will show that LMLE outperforms ROS, RUS, thresholding, and cost-sensitive learning on the CelebA data set.

Deep over-sampling (DOS)

Ando and Huang [117] introduced over-sampling to the deep feature space produced by CNNs in their DOS framework. The proposed method is extensively evaluated by generating imbalanced data sets from five popular image benchmark data sets, including MNIST, MNIST-back-rot, SVHN [124], CIFAR-10, and STL-10 [125]. Class-wise precision and recall, F1-scores, and AUC are used to evaluate the DOS framework against alternative methods.

The DOS framework consists of two simultaneous learning procedures, optimizing the lower layer and upper layer parameters separately. The lower layers are responsible for acquiring the embedding function, while the upper layers learn to discriminate between classes using the generated embeddings. In order to learn the embedding features, the CNN's input is presented with both a class label and a set of deep feature targets, an in-class nearest neighbor cluster from deep feature space. Then the *micro-cluster* loss computes the distances between each of the deep feature targets and their mean, constraining the optimization of the lower layers to shift deep feature embeddings towards the class mean. Ando and Huang state that shifting target representations to their corresponding local mean will induce smaller in-class variance and strengthen class distinction in the learned representations. The upper layers used for discriminating between classes are trained by taking the weighted sum of the CE losses, i.e. the CE loss for each deep feature target.

The deep over-sampling component is the process of selecting k in-class neighbors from deep feature space. In order to address class imbalance, the number of in-class neighbors to select should vary between classes. For example, using $k = 3$ for the

Table 14 DOS with varying imbalance [117]

Model	Reduction rate	Class	MNIST				MNISTbr				SVHN			
			Pr	Re	F1	AUC	Pr	Re	F1	AUC	Pr	Re	F1	AUC
CNN	0.90	mnr	0.98	0.93	0.96	0.99	0.31	0.76	0.43	0.68	0.62	0.77	0.55	0.78
		mjr	0.96	0.99	0.97	1.00	0.77	0.56	0.65	0.77	0.80	0.76	0.75	0.89
	0.95	mnr	0.99	0.89	0.94	0.99	0.27	0.76	0.23	0.57	0.13	0.86	0.21	0.61
		mjr	0.93	0.98	0.95	0.99	0.52	0.69	0.58	0.67	0.89	0.61	0.71	0.87
	0.99	mnr	0.65	0.98	0.77	0.96	0.31	0.71	0.43	0.68	0.50	0.72	0.42	0.74
		mjr	0.99	0.82	0.89	0.99	0.78	0.56	0.65	0.77	0.73	0.67	0.60	0.81
CNN-CL	0.90	mnr	0.99	0.90	0.94	1.00	0.22	0.77	0.31	0.69	0.59	0.60	0.42	0.78
		mjr	0.94	0.99	0.96	0.98	0.78	0.53	0.63	0.78	0.77	0.75	0.73	0.86
	0.95	mnr	0.99	0.83	0.90	0.97	0.28	0.77	0.24	0.60	0.04	0.68	0.07	0.61
		mjr	0.89	0.99	0.94	1.00	0.52	0.68	0.57	0.69	0.89	0.60	0.70	0.84
	0.99	mnr	0.75	0.98	0.85	0.95	0.22	0.72	0.31	0.69	0.47	0.71	0.37	0.72
		mjr	0.99	0.86	0.92	0.99	0.78	0.53	0.63	0.78	0.71	0.57	0.56	0.78
DOS ($k = 5$)	0.90	mnr	0.99	0.97	0.98	1.00	0.66	0.77	0.71	0.79	0.71	0.82	0.72	0.84
		mjr	0.98	0.99	0.98	1.00	0.75	0.68	0.71	0.81	0.85	0.79	0.81	0.92
	0.95	mnr	0.98	0.96	0.97	1.00	0.56	0.75	0.63	0.72	0.40	0.89	0.55	0.73
		mjr	0.97	0.99	0.98	1.00	0.64	0.74	0.69	0.78	0.90	0.69	0.78	0.91
	0.99	mnr	0.91	0.99	0.95	0.99	0.61	0.73	0.66	0.75	0.51	0.91	0.64	0.80
		mjr	0.98	0.94	0.96	1.00	0.77	0.70	0.73	0.82	0.89	0.68	0.77	0.90

Table 15 DOS with varying *k* [117]

Classifier	k	Class	MNIST				MNISTbr				SVHN			
			Pr	Re	F1	AUC	Pr	Re	F1	AUC	Pr	Re	F1	AUC
CNN		mnr	0.65	0.98	0.77	0.96	0.31	0.76	0.43	0.68	0.50	0.72	0.42	0.74
		mjr	0.99	0.82	0.89	0.99	0.78	0.56	0.65	0.77	0.73	0.67	0.60	0.81
CNN-CL		mnr	0.75	0.98	0.85	0.95	0.22	0.77	0.31	0.69	0.47	0.71	0.37	0.72
		mjr	0.99	0.86	0.92	0.99	0.78	0.53	0.63	0.78	0.71	0.57	0.56	0.78
DOS	3	mnr	0.91	0.98	0.95	0.99	0.65	0.77	0.70	0.78	0.67	0.77	0.66	0.83
		mjr	0.99	0.94	0.96	1.00	0.75	0.68	0.71	0.80	0.80	0.74	0.74	0.86
	5	mnr	0.91	0.99	0.95	0.99	0.66	0.77	0.71	0.79	0.51	0.91	0.64	0.80
		mjr	0.98	0.94	0.96	1.00	0.75	0.68	0.71	0.81	0.89	0.68	0.77	0.90
	10	mnr	0.91	0.99	0.95	0.99	0.61	0.73	0.66	0.75	0.40	0.89	0.55	0.73
		mjr	0.99	0.94	0.96	1.00	0.77	0.70	0.73	0.82	0.90	0.69	0.78	0.91

minority class, and $k = 0$ for the majority class, will supplement the minority class with additional embeddings while leaving the majority class as is.

In the first DOS experiment, they compare the DOS framework with two alternative methods for handling class imbalance, *Large Margin Local Embedding* (LMLE) [22] and *Triplet re-sampling with cost-sensitive learning* (TL-RS-CSL). MNIST back-rotation images were used to create three data sets with class reduction rates of 0%, 20%, and 40%. It was shown that the DOS framework outperformed TL-RS-CSL and LMLE for reduction rates of 20% and 40%, and that DOS performance deteriorates slower than both TL-RS-CSL and LMLE as imbalance levels increase. For example, on imbalanced MNIST-back-rot data (40% reduction), DOS scored an average class-wise recall of 75.43, compared to the LMLE score of 70.13.

In a second experiment, the DOS framework is compared to a basic CNN and a K-NN classifier that is trained with CNN deep features (CNN-CL). Three different data sets are used to generate imbalanced data sets with reduction rates of 90%, 95%, and 99%, producing imbalance ratios up to $\rho = 100$. The results in Table 14 show again that the improved performance of the DOS framework becomes more apparent as the level of class imbalance increases. The third level of imbalance, with reduction rates of 99%, show that the DOS framework clearly outperforms both the CNN and CNN-CL across all data sets and all performance metrics.

The final experiment uses balanced data sets to examine the sensitivity of DOS parameter k, which defines the number of deep feature in-class neighbors to sample. Table 15 shows that when k is increased to 10 the performance on the minority group begins to deteriorate. These results also show that the proposed DOS framework outperforms the baseline CNN and CNN-CL on balanced data sets.

The DOS method displayed no visible performance trade-offs between the majority and minority groups, or between precision and recall. Its ability to outperform alternative methods when class imbalance is not present is also very appealing, as this quality is not common among class imbalance learning methods. Most importantly, some of the performance gains observed when comparing the CNN or CNN-CL to the DOS model on the minority group were very large, e.g. MNIST-back-rot and SVHN F1-scores increasing from 0.43 to 0.66 and 0.42 to 0.64, respectively, under an imbalance reduction rate of 99%. Like other hybrid methods, DOS is more complex than more common data-level and algorithm-level methods, which may discourage statisticians from using it. We agree with Ando and Huang, however, that the DOS method is generally extendable to other domains and deep learning architectures. Future works should evaluate the DOS method in these new contexts and should compare results to other class imbalance methods presented throughout this survey.

Class rectification loss (CRL) and hard sample mining

Dong et al. [118] present an end-to-end deep learning method for addressing high class imbalance in large-scale image classification. They explicitly distinguish their work from more traditional small-scale class imbalance problems containing small levels of imbalance, suggesting that this method may not generalize to small-scale problems. Experimental results compare the proposed method against data sampling, cost-sensitive learning, threshold moving, and several relevant state-of-the-art methods. Mean sensitivity scores are used as the primary performance metric by averaging together class-wise recall scores. The proposed method combines hard sample mining from the minority groups with a regularized objective function, the *class rectification loss* (CRL).

The hard sample mining selects minority samples which are expected to be more informative for each mini-batch, allowing the model to learn more effectively with less data. Dong et al. strive to rectify, or correct, the class distribution bias in an incremental manner, resulting in progressively stronger minority class discrimination through training. Unlike LMLE [22], which attempts to strengthen the structure of both majority and minority groups, this method only enhances minority class discrimination. To identify the minority classes, all classes are sorted by their size and then selected from smallest to largest until they collectively sum up to at most half the size of the data set, ensuring

Table 16 CRL with CelebA Data (Balanced Accuracy) [118]

	Attractive	Mouth open	Smiling	Wear lipstick	High cheekbones	Male	Heavy makeup	Wavy hair	Oval face	Pointy nose	Arched eyebrows	Black hair	Big lips	Big nose	Young	Straight hair	Brown hair	Bags under eyes	Wear earrings	No beard	Bangs
Imbalance (1:x)	1	1	1	1	1	1	2	2	3	3	3	3	3	3	4	4	4	4	4	5	6
Triplet-kNN	83	92	92	91	86	91	88	77	61	61	73	82	55	68	75	63	76	63	69	82	81
PANDA	85	93	*98*	*97*	*89*	***99***	95	78	66	67	77	84	56	72	78	66	*85*	67	77	87	92
ANet	*87*	***96***	97	95	*89*	***99***	*96*	*81*	67	69	76	*90*	57	*78*	*84*	69	83	70	*83*	*93*	90
DeepID2	78	89	89	92	84	94	88	73	63	66	77	83	62	73	76	65	79	74	75	88	91
Over-sampling	77	89	90	92	84	95	87	70	63	67	*79*	84	61	73	75	66	82	73	76	88	90
Down-sampling	78	87	90	91	80	90	89	70	58	63	70	80	61	76	80	61	76	71	70	88	88
Cost-sensitive	78	89	90	91	85	93	89	75	64	65	78	85	61	74	75	67	84	74	76	88	90
Threshold-Adj	69	89	88	89	83	95	89	77	***72***	***72***	76	86	*66*	76	24	***73***	81	*76*	76	15	93
LMLE-kNN	***88***	***96***	***99***	***99***	***92***	***99***	***98***	***83***	*68*	***72***	*79*	***92***	60	***80***	***87***	***73***	***87***	73	*83*	***96***	***98***
CRL	81	*94*	92	95	87	*98*	90	79	66	*71*	***80***	88	***67***	77	83	72	84	***79***	***84***	*93*	*95*

	Blond hair	Bushy eyebrows	Wear necklace	Narrow eyes	5 o'clock shadow	Receding hairline	Wear necktie	Eyeglasses	Rosy cheeks	Goatee	Chubby	Sideburns	Blurry	Wear hat	Double chin	Pale skin	Gray hair	Mustache	Bald	Mean
Imbalance (1:x)	6	6	7	8	8	11	13	14	14	15	16	17	18	19	20	22	23	24	43	
Triplet-kNN	81	68	50	47	66	60	73	82	64	73	64	71	43	84	60	63	72	57	75	72
PANDA	91	74	51	51	76	67	85	88	68	84	65	81	50	90	64	69	79	63	74	77

Table 16 (continued)

	Blond hair	Bushy eyebrows	Wear necklace	Narrow eyes	5 o'clock shadow	Receding hairline	Wear necktie	Eyeglasses	Rosy cheeks	Goatee	Chubby	Sideburns	Blurry	Wear hat	Double chin	Pale skin	Gray hair	Mustache	Bald	Mean
ANet	90	*82*	59	57	81	70	79	95	76	86	70	79	56	90	68	77	85	61	73	80
DeepID2	90	78	70	64	*85*	81	83	92	86	*90*	81	89	74	90	83	81	90	88	93	81
Over-sampling	90	80	*71*	65	*85*	82	79	91	*90*	89	*83*	*90*	76	89	*84*	82	90	*90*	92	82
Down-sampling	85	75	66	61	82	79	80	85	82	85	78	80	68	90	80	78	88	60	79	78
Cost-sensitive	89	79	*71*	65	84	81	82	91	***92***	86	82	*90*	76	90	*84*	80	90	88	*93*	82
Threshold-Adj	92	***84***	62	*71*	82	*83*	76	95	82	89	81	89	*78*	*95*	83	*85*	*91*	86	*93*	79
LMLE-kNN	***99***	*82*	59	59	82	76	***90***	*98*	78	***95***	79	88	59	***99***	74	80	*91*	73	90	*84*
CRL	*95*	***84***	***73***	***73***	***89***	***88***	*87*	***99***	*90*	***95***	***87***	***95***	***86***	***99***	***89***	***92***	***96***	***93***	***99***	***87***

Bold-italic and italic scores indicate best and second best class scores, respectively

that the minority classes account for at most half the training batch. Both class-level and instance-level hardness metrics are considered when performing hard sample mining. At the class-level, hard positives refer to weak recognitions, the correctly labelled samples with low prediction scores. Alternatively, hard negatives are the obvious mistakes, the samples incorrectly labelled with high prediction scores. At the instance-level, hard positives are defined by correctly labelled images with far distances in feature space distance, while hard negatives are the images incorrectly labelled which are close in feature space.

The CRL loss function (Eq. 17), where $\alpha = \eta\Omega_{imb}$ is linearly proportional to the level of class imbalance, places a batch-wise class balancing constraint on the optimization process. This reduces the learning bias caused by the majority group's over-representation. The CRL regularization imposes an imbalance-adaptive learning mechanism, applying more weight to the more highly imbalanced labels, while reducing the weight for the less imbalanced labels. Three different loss criteria $\mathcal{L}_{crl}$ are explored through experimentation, where the *Triplet Ranking loss* [126] is selected as the default for many of the experiments.

$$\mathcal{L}_{bln} = \alpha\mathcal{L}_{crl} + (1-\alpha)\mathcal{L}_{ce}, \quad \alpha = \eta\Omega_{imb} \tag{17}$$

Experiments are conducted by extending state-of-the-art CNN architectures with the proposed method and performing classification on three benchmark data sets. The CelebA data set contains a max imbalance level of $\rho = 49$. The X-Domain [127] data set contains 245,467 retail store clothing images that are annotated with 9 multi-class attribute labels, 165,467 of which are set aside for training. The X-Domain contains extreme class imbalance ratios of $\rho > 4000$. Several imbalanced data sets are generated from the CIFAR-100 set, with imbalance ratios up to $\rho = 20$, to demonstrate how CRL handles increasing levels of imbalance.

Table 16 contains the CelebA facial attribute recognition results. All class imbalance methods presented were implemented on top of the 5-layer CNN, *DeepID2*, [128]. Referring to the bottom half of the data attributes, with imbalance ratios $\rho \geq 6$, under-sampling almost always performs worse than over-sampling, cost-sensitive learning, and threshold moving. According to the mean sensitivity scores, over-sampling and cost-sensitive learning perform the same, outperforming thresholding and under-sampling by 3% and 4%, respectively.

When compared to the best non-imbalanced learning method (DeepID2) and the best imbalanced learning method (LMLE), the proposed CRL method improves the mean sensitivity by 6% and 3%, respectively. By analyzing the attribute-level scores, we can see that LMLE outperforms CRL in many cases where class imbalance levels are low, e.g. outperforming CRL on the attractive attribute by 7%. Dong et al. acknowledge that LMLE appears to handle low-level imbalance better than CRL. For many of the high-imbalance attributes, the non-imbalance method DeepID2 outperforms the LMLE method. It is when class imbalance is highest that the proposed CRL method performs its best and achieves its most significant performance gains over alternative methods.

In another experiment with the X-Domain data set, CRL outperforms LMLE on all categories, achieving a mean sensitivity performance gain over LMLE of almost 5%. Cost-sensitive learning, thresholding, and ROS all score roughly 6% below CRL,

while RUS again performs very poorly. When experimenting with balanced distributions from the CIFAR-100 data set, the CRL method is applied to three state-of-the-art CNN models: CifarNet [129], ResNet32 [98], and DenseNet [130]. For each model, the CRL method consistently improves upon the baseline, increasing mean sensitivity by 3.6%, 1.2%, and 0.8%, respectively. Additional cost analysis by Dong et al. shows that CRL is significantly faster than LMLE, taking just 27.2 h to train versus LMLE's 166.6 h on the given test case.

The advantages of CRL over alternative methods on large-scale class imbalanced image data were demonstrated through comprehensive experiments. It was shown to efficiently outperform many popular class imbalanced methods, including ROS, RUS, thresholding, cost-sensitive learning, and LMLE. It was also shown to train significantly faster than the LMLE method. The authors were clear to distinguish this method as effective in large-scale image data containing high levels of imbalance, and results on facial attribute detection showed that LMLE performed better for many facial attributes with lower levels of imbalance. This suggests that the CRL method may not be a good fit for class imbalance problems containing low levels of class imbalance. Despite this initial observation, CRL should be evaluated on a wide range of data sets with varying levels of complexity to better understand when it should be used. Exploring the CRL method's ability to learn from non-image data is also of interest.

Summary of hybrid methods

This section analyzed three hybrid deep learning methods for addressing class imbalance. The LMLE method proposed by Huang et al. outperformed several state-of-the-art models on the CelebA and BSDS500 image data sets. Ando and Huang introduced the DOS method, which learns an embedding layer that produces more discriminative features, and then supplements the minority group by over-sampling in deep feature space. DOS was shown to outperform LMLE and baseline CNNs on multiple class imbalanced image data sets. Lastly, Dong et al.'s CRL loss function was shown to outperform four state-of-the-art models, ROS, RUS, cost-sensitive learning, output thresholding, and LMLE on the CelebA data set.

The results provided by Dong et al. are most informative, as they compare their CRL method to LMLE and several other data-level and algorithm-level deep learning methods for addressing class imbalance. Not only did they show that the CRL method outperforms alternative methods as class imbalance levels increase, but they also illustrated the effectiveness of RUS, ROS, thresholding, and cost-sensitive learning. From their results on the CelebA data set, we observe that ROS and cost-sensitive learning outperform RUS and threshold moving on average. These results are consistent with those from the "Data-level methods" section, suggesting that ROS is generally a good choice in addressing class imbalance with DNNs. Dong et al. also confirmed our original observation, that LMLE is rather complex and resource intensive, by showing that CRL is capable of training 6× faster than LMLE. Comparing DOS and CRL directly will prove useful, as both methods were shown to outperform the LMLE method.

In general, hybrid methods for learning from class imbalanced data will be more complex, and more difficult to implement than algorithm-level methods and data-level

Table 17 Summary of deep learning class imbalance methods

Method	Network type	Method type	Description
ROS [23, 79]	CNN	Data	ROS of minority classes until class balance is achieved
RUS [23]	CNN	Data	RUS of majority classes until class balance is achieved
Two-phase learning [20, 23]	CNN	Data	Pre-training with RUS or ROS, then fine-tuning with all data
Dynamic sampling [21]	CNN	Data	Sampling rates adjust throughout training based on previous iteration's class-wise F1-scores
MFE and MSFE loss [18]	MLP	Algorithm	New loss functions allow positive and negative classes to contribute to loss equally
Focal loss [88, 103]	CNN	Algorithm	New loss function down-weights easy-to-classify samples, reducing their impact on total loss
CSDNN [89]	MLP	Algorithm	CE loss function modified to incorporate a pre-defined cost matrix
CoSen CNN [19]	CNN	Algorithm	Cost matrix is learned through backpropagation and incorporated into output layer
CSDBN-DE [90]	DBN	Algorithm	Cost matrix is learned through evolutionary algorithm and incorporated into output layer
Threshold moving [23]	CNN	Algorithm	Decision threshold is adjusted by dividing output probabilities by prior class probabilities
Category centers [91]	CNN	Algorithm	Class centroids are calculated in deep feature space and K-NN method discriminates
Very-deep NNs [92]	CNN	Algorithm	CNN network depths of up to 50 layers are used to examine convergence rates
LMLE [22]	CNN	Hybrid	Triple-header hinge loss and quintuplet sampling generate more discriminative features
DOS [117]	CNN	Hybrid	Minority class over-sampled in deep feature space using K-NN and micro-cluster loss
CRL loss [118]	CNN	Hybrid	Class Rectification loss and hard sample mining produce more discriminative features

methods. This is expected, since both algorithm-level and data-level methods are being combined for the purpose of improving classification performance. As learners become more complex, their flexibility and ease of use will decrease, which may make them more difficult to adapt to new problems.

Class-wise performance scores on the CelebA data set in Table 16 show that different methods perform better when subjected to different levels of class imbalance. This observation supports our demand for future research that evaluates multiple deep learning methods across a variety of class imbalance levels and problem complexities. We expect to see similar trade-offs between models when they are evaluated across a variety of data types and DNN architectures. Filling these gaps in current research will help to shape future deep learning applications that involve class imbalance, class rarity, and big data.

Discussion of surveyed works

To provide a high-level summary and better compare the current deep learning methods for class imbalance, the surveyed works and their data sets have been summarized in Tables 17 and 18. The methods presented by each group have been categorized in

Table 18 Summary of data sets and class imbalance levels

Paper	Data sets	Data type	Class count	Data set size	Min class size	Max class size	ρ (Eq. 1)
[79]	CIFAR-10	Image	10	60,000	2340	3900	2.3
[20]	WHOI-Plankton	Image	103	3,400,000	< 3500	2,300,000	657
[21]	Public cameras	Image	19	10,000	14	6986	499
[18]	CIFAR-100 (1)	Image	2	6000	150	3000	20
	CIFAR-100 (2)	Image	2	1200	30	600	20
	CIFAR-100 (3)	Image	2	1200	30	600	20
	20 News Group (1)	Text	2	1200	30	600	20
	20 News Group (2)	Text	2	1200	30	600	20
[88]	COCO	Image	2	115,000	10	100,000	10,000
[103]	Building changes	Image	6	203,358	222	200,000	900
[89]	GHW	Structured	2	2565	406	2159	5.3
	ORP	Structured	2	700	124	576	4.6
[19]	MNIST	Image	10	70,000	600	6000	10
	CIFAR-100	Image	100	60,000	60	600	10
	CALTECH-101	Image	102	9144	15	30	2
	MIT-67	Image	67	6700	10	100	10
	DIL	Image	10	1300	24	331	13
	MLC	Image	9	400,000	2600	196,900	76
[90]	KEEL	Structured	2	3339	26	3313	128
[91]	CIFAR-10	Image	10	60,000	250	5000	20
	CIFAR-100	Image	100	60,000	25	500	20
[22]	CelebA	Image	2	160,000	3200	156,800	49
[117]	MNIST	Image	10	60,000	50	5000	100
	MNIST-back-rot	Image	10	62,000	12	1200	100
	CIFAR-10	Image	10	60,000	5000	5000	1
	SVHN	Image	10	99,000	73	7300	100
	STL-10	Image	10	13,000	500	500	1
[118]	CelebA	Image	2	160,000	3200	156,800	49
[92]	EmotioNet	Image	2	450,000	45	449,955	10,000
[23]	MNIST	Image	10	60,000	1	5000	5000
	CIFAR-10	Image	10	60,000	100	5000	50
	ImageNet	Image	1000	1,050,000	10	1000	100

Images from CelebA and EmotioNet are treated as a set of binary classification problems, because they are each annotated with 40 and 11 binary attributes, respectively. The COCO data class imbalance arises from the extreme imbalance between background and foreground concepts

Table 17 as one of three types: data, algorithm, or hybrid. All of the general methods for handling class imbalance in traditional machine learning have been extended to deep learning, including: random sampling, informed sampling, cost-sensitive learning, thresholding, and hybrid methods.

Data-level techniques for addressing class imbalance in deep learning include ROS [23, 79], RUS [23], two-phase learning with sampling [20, 23], and performance-based dynamic sampling [21]. The ROS experiments reported that over-sampling to the level of class balance works best on imbalanced image data. The two-phase learning method uses RUS or ROS to pre-train a DNN with class balanced data sets, then fine-tunes the network using all data. The dynamic sampling method uses class-wise F1-scores to determine sampling rates, allowing the model to sample difficult classes at a higher rate.

Algorithm-level methods were explored by nine studies, and can be further broken down into new loss functions, cost-sensitive methods, output thresholding, and a deep feature 1-NN rule. MFE and MSFE loss functions [18] are modifications of the MSE loss that allow the positive and negative class to equally contribute to the loss. The focal loss function [88, 103] improves classification by down-weighting easy-to-classify samples, preventing easily-classified negative samples from dominating the loss. Wang et al. modified the CE loss slightly to incorporate pre-defined class costs into the learning process, creating an optimization process that minimizes total cost. The final two cost-sensitive methods evaluated are unique in the sense that they iteratively improve the cost matrix throughout training, instead of requiring pre-defined costs. The CoSen CNN [19] uses an additional loss function to learn the cost parameters through backpropagation, while the CSDBN-DE [90] utilizes an evolutionary algorithm to produce increasingly better cost values. Buda et al. used class prior probabilities to adjust deep CNN output thresholds and reduce bias towards the majority group. The category centers method [91] uses CNN-generated features to calculate class centers in feature space, then uses the 1-NN rule to classify new data by its nearest class centroid. Very-deep CNNs (> 10 layers) were explored by Ding et al. to determine if deeper networks would converge faster than shallow networks when trained on imbalanced data.

Three hybrid methods that combine data-level and algorithm-level changes to address the class imbalance problem were compared to baselines and alternative methods. LMLE [22] combines quintuplet sampling with the triple-header hinge loss to learn more discriminative features. The CRL loss function combined with hard sample mining [118] was shown to improve representation learning and outperform LMLE. Ando and Huang were the first to explore over-sampling in deep feature space, showing that DOS is able to outperform LMLE on imbalanced image data.

Two-phase learning [20] and dynamic sampling [21] both outperformed transfer learning and augmentation methods in classifying imbalanced image data. Buda et al. presented conflicting results, however, that suggested that plain ROS performs better than both two-phase learning and RUS. Dong et al. showed that ROS and cost-sensitive learning methods perform equivalently on the CelebA dataset, both outperforming RUS and thresholding methods. Experiments by Khan et al., evaluated over six data sets, showed that a cost-sensitive CNN can outperform data sampling methods with neural networks and cost-sensitive SVM and RF learners. Wang et al. deployed a state-of-the-art hospital readmission predictor built using a cost-sensitive DNN that outperformed existing models. A class balanced version of the MSE loss, MFE and MSFE [18], showed improvements in classifying imbalanced image and text data. In [117], over-sampling in the deep feature space is used to perform classification on five data sets, showing that the DOS framework can outperform LMLE and TL-RS-CSL. The focal loss [88] function was shown to outscore leading one-stage and two-stage detectors on the COCO data set. The CRL loss [118] outperformed LMLE, data sampling, cost-sensitive learning, and thresholding on the CelebA facial attribute detection task.

Multiple authors [19, 23, 79] suggested the use of deep learning with ROS to address class imbalance, showing that ROS outperforms RUS in almost all cases. Buda et al. and Hensman and Masko both suggest applying ROS until class imbalance is eliminated in the training set, but experiments did not consider big data scenarios. We

believe that applying ROS to this level will not hold when subjected to big data or class rarity. Duplicating such large volumes of data may cause over-fitting and is very computationally expensive. In a contrastive example, Bauder et al. [131] found RUS to be more effective than ROS when using traditional machine learning algorithms to detect fraud in big data with class rarity. Future work should consider these big data scenarios, and should experiment with RUS methods that remove duplicates and strengthen class boundaries.

Almost all (80%) of the surveyed works employed deep CNN architectures to address class imbalanced image data. Yet, all of the methods presented can be extended to non-CNN architectures and non-image data. The popularity of CNNs, image classification, and object detection in the research community can be partly attributed to popular benchmark data sets like MNIST and CIFAR, and the continuous improvements being driven by competitive events like LSVRC. Class imbalance is not limited to image data, and additional work must be done to evaluate the use of these deep learning class imbalance methods in other domains.

Approximately half of the studies used only one data set when evaluating their deep learning methods for addressing class imbalance. The authors are not at fault for this, as most were focused on solving a specific problem or benchmark task. However, more comprehensive studies that evaluate these methods across a wider range of data sets, with varying levels of class imbalance and complexity, will better demonstrate their strengths and weaknesses. Also, only a third of the studies indicate the number of rounds or repetitions executed for each experiment. In other words, the remaining groups either did not perform multiple runs, or failed to include those details and presented the most favorable results. To their defense, training a deep learning model on a large data set can take days or even weeks, making it difficult to conduct several rounds of experiments. This creates several avenues for future research, as comparing various deep learning methods on many data sets with repetition will increase confidence in results and help guide future practitioners in model selection.

Less than half of the surveyed works experimented with data sets larger than 100,000 samples. Lee et al. were the only ones to work with more than a million samples containing imbalanced data. Pouyanfar et al. provided a big data analytics workflow to classify fast-streaming network camera data, but their experiment was limited to just 10,000 images. This is an extremely important area of research that demands attention if we expect deep learning to provide big data analytics solutions. It is also very likely that the methods proposed throughout this survey, e.g. ROS and RUS, will behave very differently in the context of big data.

It is not currently possible to compare the methods which have been presented directly, as they are evaluated across a variety of data sets with varying levels of class imbalance, and results are reported with inconsistent performance metrics. In addition, some studies report contradictory results, further suggesting that performance is highly dependent on problem complexity, class representation, and the performance metrics reported. Overall, there is a lack of evidence which distinguishes any one deep learning method as superior for learning from class imbalanced data, and additional experiments are required before such conclusions can be made.

Conclusion

To the best of our knowledge, this survey provides the most comprehensive analysis of deep learning methods for addressing the class imbalance data problem. Fifteen studies published between 2015 and 2018 are summarized and discussed, exploring a number of advanced techniques for learning from imbalanced data with DNNs. It has been shown that traditional machine learning techniques for handling class imbalance can be extended to deep learning models with success. The survey also finds that nearly all research in this area has been focused on computer vision tasks with CNNs. Despite a growing demand for big data analytics solutions, there is very little research that properly evaluates deep learning in the context of class imbalance and big data. Deep learning from class imbalanced data is still largely understudied, and statistical evidence which compares newly published methods across a variety of data sets and imbalance levels does not exist.

Several areas for future work are apparent. Applying the newly proposed methods to a larger variety of data sets and class imbalance levels, comparing results with multiple complementary performance metrics, and reporting statistical evidence will help to identify the preferred deep learning method for future applications containing class imbalance. Experimenting with deep learning methods for addressing class imbalance in the context of big data and class rarity will prove valuable to the future of big data analytics. More work is required with non-convolutional DNNs to determine if the methods presented will generalize well to alternative architectures, e.g. MLPs and RNNs. Finally, research that evaluates the use of deep learning to address class imbalance in non-image data is limited and should be expanded upon.

Abbreviations

ANN: artificial neural network; AP: average precision; AUC: area under the curve; C2C: class-to-class; CE: cross entropy; CNN: convolutional neural network; CRL: class rectification loss; CSDBN-DE: cost-sensitive deep belief network with differential evolution; CSDNN: cost-sensitive deep neural network; DBN: deep belief network; DNN: deep neural network; DOS: deep over-sampling; ELM: extreme learning machine; FAU: Facial Action Unit; FL: focal loss; FN: false negative; FNE: false negative error; FP: false positive; FPE: false positive error; FPN: feature pyramid network; GHW: general hospital wards; GPU: graphics processing unit; IoT: internet of things; K-NN: K-nearest neighbors; KEEL: Knowledge Extraction based on Evolutionary Learning; LMLE: Large Margin Local Embedding; LSVRC: Large Scale Visual Recognition Challenge; MFE: mean false error; MLP: multilayer perceptron; MSE: mean squared error; MSFE: mean squared false error; NPV: negative predictive value; OHEM: online hard example mining; ORP: operating room pilot; PPV: positive predictive value; PR: precision–recall; ReLU: rectified linear unit; RF: random forest; RNN: recurrent neural network; ROC: receiver operating characteristics; ROS: random over-sampling; RUS: random under-sampling; SMOTE: Synthetic Minority Over-sampling Technique; SVM: support vector machine; TL-RS-CSL: triplet re-sampling with cost-sensitive learning; TN: true negative; TNR: true negative rate; TP: true positive; TPR: true positive rate.

Authors' contributions

JMJ performed the literature review and drafted the manuscript. TMK worked with JMJ to develop the article's framework and focus. TMK introduced this topic to JMJ. Both authors read and approved the final manuscript.

Acknowledgements

The authors would like to thank the anonymous reviewers for their constructive evaluation of this paper, and the various members of the Data Mining and Machine Learning Laboratory, Florida Atlantic University, for assistance with the reviews.

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

Not applicable.

Consent for publication

Not applicable.

Ethics approval and consent to participate

Not applicable.

Funding
Not applicable.

Publisher's Note

Received: 9 January 2019 Accepted: 8 March 2019
Published online: 19 March 2019

References

1. Witten IH, Frank E, Hall MA, Pal CJ. Data mining, Fourth Edition: Practical machine learning tools and techniques. 4th ed. San Francisco: Morgan Kaufmann Publishers Inc.; 2016.
2. Kotsiantis SB. Supervised machine learning: a review of classification techniques. In: Proceedings of the 2007 conference on emerging artificial intelligence applications in computer engineering: Real Word AI Systems with applications in eHealth, HCI, Information Retrieval and Pervasive Technologies. IOS Press, Amsterdam, The Netherlands, The Netherlands; 2007. p. 3–24. http://dl.acm.org/citation.cfm?id=1566770.1566773. Accessed 15 Oct 2018.
3. Rao RB, Krishnan S, Niculescu RS. Data mining for improved cardiac care. SIGKDD Explor Newsl. 2006;8(1):3–10. https://doi.org/10.1145/1147234.1147236.
4. Wei W, Li J, Cao L, Ou Y, Chen J. Effective detection of sophisticated online banking fraud on extremely imbalanced data. World Wide Web. 2013;16(4):449–75. https://doi.org/10.1007/s11280-012-0178-0.
5. Herland M, Khoshgoftaar TM, Bauder RA. Big data fraud detection using multiple medicare data sources. J Big Data. 2018;5(1):29. https://doi.org/10.1186/s40537-018-0138-3.
6. Cieslak DA, Chawla NV, Striegel A. Combating imbalance in network intrusion datasets. In: 2006 IEEE international conference on granular computing. 2006. p. 732–7. https://doi.org/10.1109/GRC.2006.1635905.
7. Kubat M, Holte RC, Matwin S. Machine learning for the detection of oil spills in satellite radar images. Mach Learn. 1998;30(2):195–215. https://doi.org/10.1023/A:1007452223027.
8. Bauder RA, Khoshgoftaar TM, Hasanin T. An empirical study on class rarity in big data. In: 2018 17th IEEE international conference on machine learning and applications (ICMLA). 2018. p. 785–90. https://doi.org/10.1109/ICMLA.2018.00125.
9. Bauder RA, Khoshgoftaar TM. The effects of varying class distribution on learner behavior for medicare fraud detection with imbalanced big data. Health Inf Sci Syst. 2018;6(1):9. https://doi.org/10.1007/s13755-018-0051-3.
10. Krawczyk B. Learning from imbalanced data: open challenges and future directions. Prog Artif Intell. 2016;5(4):221–32. https://doi.org/10.1007/s13748-016-0094-0.
11. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521:436.
12. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M, Ghemawat S, Goodfellow I, Harp A, Irving G, Isard M, Jia Y, Jozefowicz R, Kaiser L, Kudlur M, Levenberg J, Mané D, Monga R, Moore S, Murray D, Olah C, Schuster M, Shlens J, Steiner B, Sutskever I, Talwar K, Tucker P, Vanhoucke V, Vasudevan V, Viégas F, Vinyals O, Warden P, Wattenberg M, Wicke M, Yu Y, Zheng X. TensorFlow: large-scale machine learning on heterogeneous systems 2015. http://tensorflow.org/. Accessed 1 Nov 2018.
13. Theano Development Team. Theano: a Python framework for fast computation of mathematical expressions. 2016. arXiv e-prints arXiv:1605.02688.
14. Chollet F, et al. Keras. 2015. https://keras.io. Accessed 1 Nov 2018.
15. Paszke A, Gross S, Chintala S, Chanan G, Yang E, DeVito Z, Lin Z, Desmaison A, Antiga L, Lerer A. Automatic differentiation in pytorch. In: NIPS-W. 2017.
16. Chetlur S, Woolley C, Vandermersch P, Cohen J, Tran J, Catanzaro B, Shelhamer, E. cudnn: Efficient primitives for deep learning 2014.
17. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Neural information processing systems. 2012. p. 25. https://doi.org/10.1145/3065386.
18. Wang S, Liu W, Wu J, Cao L, Meng Q, Kennedy PJ. Training deep neural networks on imbalanced data sets. In: 2016 international joint conference on neural networks (IJCNN). 2016. p. 4368–74. https://doi.org/10.1109/IJCNN.2016.7727770.
19. Khan SH, Hayat M, Bennamoun M, Sohel FA, Togneri R. Cost-sensitive learning of deep feature representations from imbalanced data. IEEE Trans Neural Netw Learn Syst. 2018;29:3573–87.
20. Lee H, Park M, Kim J. Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning. In: 2016 IEEE international conference on image processing (ICIP). 2016. p. 3713–7. https://doi.org/10.1109/ICIP.2016.7533053.
21. Pouyanfar S, Tao Y, Mohan A, Tian H, Kaseb AS, Gauen K, Dailey R, Aghajanzadeh S, Lu Y, Chen S, Shyu M. Dynamic sampling in convolutional neural networks for imbalanced data classification. In: 2018 IEEE conference on multimedia information processing and retrieval (MIPR). 2018. p. 112–7. https://doi.org/10.1109/MIPR.2018.00027.
22. Huang C, Li Y, Loy CC, Tang X. Learning deep representation for imbalanced classification. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR). 2016. p. 5375–84. https://doi.org/10.1109/CVPR.2016.580.
23. Buda M, Maki A, Mazurowski MA. A systematic study of the class imbalance problem in convolutional neural networks. Neural Netw. 2018;106:249–59. https://doi.org/10.1016/j.neunet.2018.07.011.
24. Brocke J.v, Simons A, Niehaves B, Riemer K, Plattfaut R, Cleven A. Reconstructing the giant: on the importance of rigour in documenting the literature search process. 2009.
25. Google Scholar. https://scholar.google.com/. Accessed 15 Nov 2018.
26. IEEE Xplore Digital Library. https://ieeexplore.ieee.org. Accessed 15 Nov 2018.

27. Wang S, Yao X. Multiclass imbalance problems: analysis and potential solutions. IEEE Trans Syst Man Cybern Part B (Cybern). 2012;42(4):1119–30. https://doi.org/10.1109/TSMCB.2012.2187280.
28. He H, Garcia EA. Learning from imbalanced data. IEEE Trans Knowl Data Eng. 2009;21(9):1263–1284. https://doi.org/10.1109/TKDE.2008.239.
29. Japkowicz N. The class imbalance problem: Significance and strategies. In: In proceedings of the 2000 international conference on artificial intelligence (ICAI). 2000;111–7.
30. Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A. Mining data with rare events: a case study. In: Proceedings of the 19th IEEE international conference on tools with artificial intelligence—Vol. 02. ICTAI '07, IEEE Computer Society, Washington, DC. 2007. p. 132–9. https://doi.org/10.1109/ICTAI.2007.130.
31. Weiss GM. Mining with rarity: a unifying framework. SIGKDD Explor Newsl. 2004;6(1):7–19. https://doi.org/10.1145/1007730.1007734.
32. Provost F, Fawcett T. Analysis and visualization of classifier performance: comparison under imprecise class and cost distributions. In: Proceedings of the third international conference on knowledge discovery and data mining. 1999. p. 43–8.
33. Weng C.G, Poon J. A new evaluation measure for imbalanced datasets. In: Proceedings of the 7th Australasian data mining conference—Vol. 87. AusDM '08. Australian Computer Society, Inc., Darlinghurst, Australia. 2008. p. 27–32. http://dl.acm.org/citation.cfm?id=2449288.2449295.
34. Davis J, Goadrich M. The relationship between precision-recall and roc curves. In: Proceedings of the 23rd international conference on machine learning. ICML '06. ACM, New York, NY, USA. 2006. p. 233–40. https://doi.org/10.1145/1143844.1143874.
35. Seliya N, Khoshgoftaar T.M, Van Hulse J. A study on the relationships of classifier performance metrics. In: 2009 21st IEEE international conference on tools with artificial intelligence. 2009. p. 59–66. https://doi.org/10.1109/ICTAI.2009.25.
36. Brier GW. Verification of forecasts expressed in terms of probability. Mon Weather Rev. 1950;78(1):1–3. https://doi.org/10.1175/1520-0493(1950)078<0001:VOFEIT>2.0.CO;2 .
37. Van Hulse J, Khoshgoftaar TM, Napolitano A. Experimental perspectives on learning from imbalanced data. In: Proceedings of the 24th international conference on machine learning. ICML '07. ACM, New York, NY, USA. 2007. p. 935–42. https://doi.org/10.1145/1273496.1273614.
38. Chawla NV, Japkowicz N, Kotcz A. Editorial: Special issue on learning from imbalanced data sets. SIGKDD Explor Newsl. 2004;6(1):1–6. https://doi.org/10.1145/1007730.1007733.
39. Zhang J, Mani I. KNN approach to unbalanced data distributions: a case study involving information extraction. In: Proceedings of the ICML'2003 workshop on learning from imbalanced datasets. 2003.
40. Kubat M, Matwin S. Addressing the curse of imbalanced training sets: one-sided selection. In: Fourteenth international conference on machine learning. 2000.
41. Tomek I. Two modifications of cnn. IEEE Trans Syst Man Cybern. 1976;6:769–72.
42. Barandela R, Valdovinos RM, Sánchez JS, Ferri FJ. The imbalanced training sample problem: under or over sampling? In: Fred A, Caelli TM, Duin RPW, Campilho AC, de Ridder D, editors. Structural, syntactic, and statistical pattern recognition. Berlin: Springer; 2004. p. 806–14.
43. Wilson DL. Asymptotic properties of nearest neighbor rules using edited data. IEEE Trans Syst Man Cybern SMC. 1972;2(3):408–21. https://doi.org/10.1109/TSMC.1972.4309137.
44. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. Smote: synthetic minority over-sampling technique. J Artif Int Res. 2002;16(1):321–57.
45. Han H, Wang W-Y, Mao B-H. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In: Huang D-S, Zhang X-P, Huang G-B, editors. Adv Intell Comput. Berlin: Springer; 2005. p. 878–87.
46. Bunkhumpornpat C, Sinapiromsaran K, Lursinsap C. Safe-level-smote: safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In: Theeramunkong T, Kijsirikul B, Cercone N, Ho T-B, editors. Advances in knowledge discovery and data mining. Berlin, Heidelberg: Springer; 2009. p. 475–82.
47. Holte RC, Acker L, Porter BW. Concept learning and the problem of small disjuncts. In: IJCAI. 1989.
48. Weiss G, Hirsh H. A quantitative study of small disjuncts. In: In proceedings of the seventeenth national conference on artificial intelligence and twelfth conference on innovative applications of artificial intelligence. 2000. p. 665–70.
49. Jo T, Japkowicz N. Class imbalances versus small disjuncts. SIGKDD Explor Newsl. 2004;6(1):40–9. https://doi.org/10.1145/1007730.1007737.
50. Elkan C. The foundations of cost-sensitive learning. In: In Proceedings of the seventeenth international joint conference on artificial intelligence. 2001. p. 973–8.
51. Ling CX, Sheng VS. In: Sammut C, editor. Cost-sensitive learning and the class imbalanced problem. 2007.
52. Liu X, Wu J, Zhou Z. Exploratory undersampling for class-imbalance learning. IEEE Trans Syst Man Cybern Part B (Cybern). 2009;39(2):539–50. https://doi.org/10.1109/TSMCB.2008.2007853.
53. Chawla NV, Lazarevic A, Hall LO, Bowyer KW. Smoteboost: improving prediction of the minority class in boosting. In: Lavrač N, Gamberger D, Todorovski L, Blockeel H, editors. Knowledge discovery in databases: PKDD 2003. Berlin, Heidelberg: Springer; 2003. p. 107–19.
54. Guo H, Viktor HL. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. SIGKDD Explor Newsl. 2004;6(1):30–9. https://doi.org/10.1145/1007730.1007736.
55. Mease D, Wyner AJ, Buja A. Boosted classification trees and class probability/quantile estimation. J Mach Learn Res. 2007;8:409–39.
56. Sun Y. Cost-sensitive boosting for classification of imbalanced data. Ph.D. thesis, Waterloo, Ont., Canada, Canada. 2007. AAINR34548.
57. Vázquez F. Deep Learning made easy with Deep Cognition. 2017. https://becominghuman.ai/deep-learning-made-easy-with-deep-cognition-403fbe445351. Accessed 10 Oct 2018.
58. Lecun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proc IEEE. 1998;86(11):2278–324.

59. Fukushima K. Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol Cybern. 1980;36(4):193–202. https://doi.org/10.1007/BF00344251.
60. Goodfellow I, Bengio Y, Courville A. Deep learning. Cambridge: The MIT Press; 2016.
61. Minar MR, Naher J. Recent advances in deep learning: an overview. 2018. arXiv:1807.08169.
62. Schmidhuber J. Deep learning in neural networks: an overview. Neural Netw. 2015;61:85–117.
63. Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature. 1986;323:533.
64. LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD. Backpropagation applied to handwritten zip code recognition. Neural Comput. 1989;1(4):541–51. https://doi.org/10.1162/neco.1989.1.4.541.
65. Hinton GE, Osindero S, Teh Y-W. A fast learning algorithm for deep belief nets. Neural Comput. 2006;18(7):1527–54. https://doi.org/10.1162/neco.2006.18.7.1527.
66. Bengio Y, Lamblin P, Popovici D, Larochelle H. Greedy layer-wise training of deep networks. In: Proceedings of the 19th international conference on neural information processing systems. NIPS'06. MIT Press, Cambridge, MA, USA. 2006. p. 153–60. http://dl.acm.org/citation.cfm?id=2976456.2976476.
67. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC, Fei-Fei L. ImageNet large scale visual recognition challenge. Int J Comput Vision (IJCV). 2015;115(3):211–52. https://doi.org/10.1007/s11263-015-0816-y.
68. Shatnawi A, Al-Ayyoub M, Albdour G, Al-Qurran R. A comparative study of open source deep learning frameworks. https://doi.org/10.1109/IACS.2018.8355444.
69. Kumar M. An incorporation of artificial intelligence capabilities in cloud computing. Int J Eng Comput Sci. 2016. https://doi.org/10.18535/ijecs/v5i11.63.
70. Saiyeda A, Mir MA. Cloud computing for deep learning analytics: a survey of current trends and challenges. Int J Adv Res Comput Sci. 2017;8(2):68–72. https://doi.org/10.26483/ijarcs.v8i2.2931.
71. Elgendy N, Elragal A. Big data analytics: a literature review paper. In: Perner P, editor. Advances in data mining. Applications and theoretical aspects. Cham: Springer; 2014. p. 214–27.
72. Dumbill E. What is big data? : an introduction to the big data landscape. 2012. http://radar.oreilly.com/2012/01/what-is-big-data.html
73. Ahmed SE. Perspectives on Big Data analysis: methodologies and applications. Providence: American Mathematical Society; 2014.
74. Leevy JL, Khoshgoftaar TM, Bauder RA, Seliya N. A survey on addressing high-class imbalance in big data. J Big Data. 2018;5(1):42. https://doi.org/10.1186/s40537-018-0151-6.
75. Najafabadi MM, Villanustre F, Khoshgoftaar TM, Seliya N, Wald R, Muharemagic E. Deep learning applications and challenges in big data analytics. J Big Data. 2015;2(1):1. https://doi.org/10.1186/s40537-014-0007-7.
76. Hinton G, Salakhutdinov R. Discovering binary codes for documents by learning deep generative models. Top Cogn Sci. 2011;3(1):74–91.
77. Salakhutdinov R, Hinton G. Semantic hashing. Int J Approx Reason. 2009;50(7):969–78. https://doi.org/10.1016/j.ijar.2008.11.006.
78. Anand R, Mehrotra KG, Mohan CK, Ranka S. An improved algorithm for neural network classification of imbalanced training sets. IEEE Trans Neural Netw. 1993;4(6):962–9. https://doi.org/10.1109/72.286891.
79. Hensman P, Masko D. The impact of imbalanced training data for convolutional neural networks. 2015.
80. Krizhevsky A, Nair V, Hinton G. Cifar-10 (Canadian Institute for Advanced Research).
81. Orenstein EC, Beijbom O, Peacock E, Sosik H. Whoi-plankton—a large scale fine grained visual recognition benchmark dataset for plankton classification. CoRR, 2015.
82. Havaei M, Davy A, Warde-Farley D, Biard A, Courville A, Bengio Y, Pal C, Jodoin P-M, Larochelle H. Brain tumor segmentation with deep neural networks. Med Image Anal. 2017;35:18–31. https://doi.org/10.1016/j.media.2016.05.004.
83. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision. In: IEEE conference on computer vision and pattern recognition (CVPR). vol. 2016. 2016. p. 2818–26.
84. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L. ImageNet: a large-scale hierarchical image database. In: CVPR09. 2009.
85. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. CoRR. 2014. arXiv:1409.1556.
86. LeCun Y, Cortes C. MNIST handwritten digit database. 2010. http://yann.lecun.com/exdb/mnist/. Accessed 15 Nov 2018.
87. Springenberg JT, Dosovitskiy A, Brox T, Riedmiller MA. Striving for simplicity: the all convolutional net. CoRR. 2014. arXiv:1412.6806.
88. Lin T-Y, Goyal P, Girshick RB, He K, Dollár P. Focal loss for dense object detection. In: IEEE international conference on computer vision (ICCV). vol. 2017. 2017. p. 2999–3007.
89. Wang H, Cui Z, Chen Y, Avidan M, Abdallah AB, Kronzer A. Predicting hospital readmission via cost-sensitive deep learning. IEEE/ACM transactions on computational biology and bioinformatics. 2018. p. 1–1. https://doi.org/10.1109/TCBB.2018.2827029.
90. Zhang C, Tan KC, Ren R. Training cost-sensitive deep belief networks on imbalance data problems. In: 2016 international joint conference on neural networks (IJCNN). 2016. p. 4362–7. https://doi.org/10.1109/IJCNN.2016.7727769.
91. Zhang Y, Shuai L, Ren Y, Chen H. Image classification with category centers in class imbalance situation. In: 2018 33rd Youth Academic annual conference of Chinese Association of Automation (YAC). 2018. p. 359–63. https://doi.org/10.1109/YAC.2018.8406400.
92. Ding W, Huang D, Chen Z, Yu X, Lin W. Facial action recognition using very deep networks for highly imbalanced class distribution. In: 2017 Asia-Pacific signal and information processing association annual summit and conference (APSIPA ASC). 2017. p. 1368–72. https://doi.org/10.1109/APSIPA.2017.8282246.
93. Krizhevsky A, Nair V, Hinton G. Cifar-100 (Canadian Institute for Advanced Research).
94. 20 Newsgroups Dataset. http://people.csail.mit.edu/jrennie/20Newsgroups/. Accessed 15 Oct 2018.

95. Girshick RB, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE conference on computer vision and pattern recognition. Vol. 2014. 2014. p. 580–7.
96. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C-Y, Berg AC. Ssd: Single shot multibox detector. In: Leibe B, Matas J, Sebe N, Welling M, editors. Computer Vision-ECCV 2016. Cham: Springer; 2016. p. 21–37.
97. Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: unified, real-time object detection. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR). 2016. p. 779–88. https://doi.org/10.1109/CVPR.2016.91.
98. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: IEEE conference on computer vision and pattern recognition (CVPR), vol. 2016. 2016. p. 770–8.
99. Lin T-Y, Maire M, Belongie SJ, Bourdev LD, Girshick RB, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL. Microsoft coco: common objects in context. In: ECCV. 2014.
100. Fu C, Liu W, Ranga A, Tyagi A, Berg AC. DSSD: deconvolutional single shot detector. CoRR. 2017. arXiv:1701.06659.
101. Shrivastava A, Sukthankar R, Malik J, Gupta A. Beyond skip connections: top-down modulation for object detection. CoRR. 2016. arXiv:1612.06851.
102. Shrivastava A, Gupta A, Girshick RB. Training region-based object detectors with online hard example mining. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR). 2016. p. 761–9.
103. Nemoto K, Hamaguchi R, Imaizumi T, Hikosaka S. Classification of rare building change using cnn with multi-class focal loss. In: IGARSS 2018—2018 IEEE international geoscience and remote sensing symposium. 2018. p. 4663–6. https://doi.org/10.1109/IGARSS.2018.8517563.
104. Li FF, Andreetto M, Ranzato MA. Caltech101 image dataset. 2003.
105. Quattoni A, Torralba A. Recognizing indoor scenes. In: 2009 IEEE conference on computer vision and pattern recognition(CVPR). 2009. p. 413–20. https://doi.org/10.1109/CVPRW.2009.5206537.
106. The University of Edinburgh: Edinburgh Dermofit Image Library. https://licensing.eri.ed.ac.uk/i/software/dermofit-image-library.html. Accessed 5 Nov 2018.
107. Beijbom O, Edmunds PJ, Kline DI, Mitchell BG, Kriegman D. Automated annotation of coral reef survey images. In: IEEE conference on computer vision and pattern recognition (CVPR), Providence, Rhode Island. 2012.
108. Chung YA, Lin HT, Yang SW. Cost-aware pre-training for multiclass cost-sensitive deep learning. In: IJCAI. 2016.
109. Ramentol E, Caballero Y, Bello R, Herrera F. Smote-rsb*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using smote and rough sets theory. Knowl Inf Syst. 2012;33(2):245–65. https://doi.org/10.1007/s10115-011-0465-6.
110. Qiu X, Xu J, Tan KC, Abbass HA. Adaptive cross-generation differential evolution operators for multiobjective optimization. IEEE Trans Evol Comput. 2016;20(2):232–44. https://doi.org/10.1109/TEVC.2015.2433672.
111. Alcalá-Fdez J, Sánchez L, García S, del Jesus MJ, Ventura S, Garrell JM, Otero J, Romero C, Bacardit J, Rivas VM, Fernández JC, Herrera F. Keel: a software tool to assess evolutionary algorithms for data mining problems. Soft Comput. 2009;13(3):307–18. https://doi.org/10.1007/s00500-008-0323-y.
112. Huang G-B, Zhu Q-Y, Siew C. Extreme learning machine: theory and applications. Neurocomputing. 2006;70:489–501. https://doi.org/10.1016/j.neucom.2005.12.126.
113. Choromanska A, Henaff M, Mathieu M, Arous GB, LeCun Y. The loss surfaces of multilayer networks. In: AISTATS. 2015.
114. Kawaguchi K. Deep learning without poor local minima. In: Lee DD, Sugiyama M, Luxburg UV, Guyon I, Garnett R, editors. Advances in neural information processing systems 29. Red Hook: Curran Associates Inc; 2016. p. 586–94.
115. Ekman P, Friesen W. Facial action coding system: a technique for the measurement of facial movement. Palo Alto: Consulting Psychologists Press; 1978.
116. Benitez-Quiroz CF, Srinivasan R, Feng Q, Wang Y, Martínez AM. Emotionet challenge: recognition of facial expressions of emotion in the wild. CoRR. 2017. arXiv:1703.01210.
117. Ando S, Huang CY. Deep over-sampling framework for classifying imbalanced data. In: Ceci M, Hollmén J, Todorovski L, Vens C, Džeroski S, editors. Machine learning and knowledge discovery in databases. Cham: Springer; 2017. p. 770–85.
118. Dong Q, Gong S, Zhu X. Imbalanced deep learning by minority class incremental rectification. In: IEEE transactions on pattern analysis and machine intelligence. 2018. p. 1–1 https://doi.org/10.1109/TPAMI.2018.2832629.
119. Liu Z, Luo P, Wang X, Tang X. Deep learning face attributes in the wild. In: Proceedings of international conference on computer vision (ICCV). 2015.
120. Schroff F, Kalenichenko D, Philbin J. Facenet: a unified embedding for face recognition and clustering. In: IEEE conference on computer vision and pattern recognition (CVPR). vol. 2015. 2015. p. 815–23.
121. Zhang N, Paluri M, Ranzato M, Darrell T, Bourdev L. Panda: pose aligned networks for deep attribute modeling. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition. 2013. https://doi.org/10.1109/CVPR.2014.212.
122. Liu Z, Luo P, Wang X, Tang X. Deep learning face attributes in the wild. In: 2015 IEEE international conference on computer vision (ICCV). 2015. p. 3730–8. https://doi.org/10.1109/ICCV.2015.425.
123. Arbelaez P, Maire M, Fowlkes C, Malik J. Contour detection and hierarchical image segmentation. IEEE Trans Pattern Anal Mach Intell. 2011;33(5):898–916. https://doi.org/10.1109/TPAMI.2010.161.
124. Netzer Y, Wang T, Coates A, Bissacco A, Wu B, Ng AY. Reading digits in natural images with unsupervised feature learning. In: NIPS workshop on deep learning and unsupervised feature Learning 2011. 2011.
125. Coates A, Ng A, Lee H. An analysis of single-layer networks in unsupervised feature learning. In: Gordon, G., Dunson, D., Dudík, M, editors. Proceedings of the fourteenth international conference on artificial intelligence and statistics. Proceedings of machine learning research. Vol. 15. PMLR, Fort Lauderdale, FL, USA. 2011. p. 215–23.
126. Liu T-Y. Learning to rank for information retrieval. Found Trends Inf Retr. 2009;3(3):225–331. https://doi.org/10.1561/1500000016.
127. Chen Q, Huang J, Feris R, Brown LM, Dong J, Yan S. Deep domain adaptation for describing people based on fine-grained clothing attributes. In: 2015 IEEE conference on computer vision and pattern recognition (CVPR). 2015. p. 5315–24. https://doi.org/10.1109/CVPR.2015.7299169.

128. Sun Y, Wang X, Tang X. Deep learning face representation by joint identification-verification. In: NIPS. 2014.
129. Krizhevsky A. Learning multiple layers of features from tiny images. University of Toronto. 2012.
130. Huang G, Liu Z, Weinberger KQ. Densely connected convolutional networks. CoRR. 2016. arXiv:1608.06993.
131. Bauder RA, Khoshgoftaar TM, Hasanin T. Data sampling approaches with severely imbalanced big data for medicare fraud detection. In: 2018 IEEE 30th international conference on tools with artificial intelligence (ICTAI). 2018. p. 137–42. https://doi.org/10.1109/ICTAI.2018.00030.