Deep Over-sampling Framework for Classifying Imbalanced Data

Shin Ando[1] and Chun Yuan Huang[2]

[1] School of Management,
Tokyo University of Science,
1-11-2 Fujimi, Chiyoda-ku, Tokyo, Japan
ando@rs.tus.ac.jp
[2] School of Management,
Tokyo University of Science,
1-11-2 Fujimi, Chiyoda-ku, Tokyo, Japan
8613095@ed.tus.ac.jp

Abstract. Class imbalance is a challenging issue in practical classification problems for deep learning models as well as traditional models. Traditionally successful countermeasures such as synthetic over-sampling have had limited success with complex, structured data handled by deep learning models. In this paper, we propose *Deep Over-sampling* (DOS), a framework for extending the synthetic over-sampling method to the deep feature space acquired by a convolutional neural network (CNN). Its key feature is an explicit, supervised representation learning, for which the training data presents each raw input sample with a synthetic embedding target in the deep feature space, which is sampled from the linear subspace of in-class neighbors. We implement an iterative process of training the CNN and updating the targets, which induces smaller in-class variance among the embeddings, to increase the discriminative power of the deep representation. We present an empirical study using public benchmarks, which shows that the DOS framework not only counteracts class imbalance better than the existing method, but also improves the performance of the CNN in the standard, balanced settings.

Keywords: Class Imbalance, Convolutional Neural Network, Deep Learning, Representation Learning, Synthetic Over-sampling

1 Introduction

In recent years, deep learning models have contributed to significant advances in supervised and unsupervised learning tasks on complex data, such as speech and imagery. Convolutional neural networks (CNNs), for example, have achieved *state-of-the-art* performances on various image classification benchmarks [27, 8]. One of the key features of CNN is representation learning, i.e., the hidden layers of convolutional neural network generates an expressive, non-linear mapping of complex data in a *deep feature* space [19, 12]. Such features are shown to

be useful for other classification models or similar classification tasks [2, 13], enabling further means of enhancements such as multi-task learning [7].

The applications of deep learning, meanwhile, encounter many practical challenges, such as the cost of preparing a sufficient amount of labeled samples. The problem of class imbalance arises when the number of samples significantly differ between two or more classes. Such imbalance can affect the traditional classification models [11, 1, 17] as well as deep learning models [14, 15, 28], commonly resulting in poor performances over the classes in the minority. For deep learning models, its influence on representation learning can deteriorate the performances on majority classes as well.

There is a rich literature on countermeasures against class imbalance for traditional classification models [4]. A popular and intuitive approach among them is re-sampling, which directly adjusts the sample sizes of respective classes. For example, SMOTE [3] generates synthetic samples, which are interpolations of *in-class* neighboring samples, to augment the minority class. Its underlying assumption is that the interpolations do not deviate from the original class distribution, as in a locally linear feature space. Similar approaches for deep learning models, e.g., re-sampling, cost-sensitive learning, and their combinations [14, 28], have also been explored, but in a more limited number of studies. Overall, they introduce complex architectures or sampling schemes, which require significant amount of data- and model-specific configurations and lower the applicability to new problems. It also should be noted that generating synthetic samples of structured, complex data, in order to conduct synthetic over-sampling on such data, is not straightforward.

In this work, we extend the synthetic over-sampling method to the convolutional neural network (CNN) using its deep representation. To our knowledge, over-sampling in the acquired, deep feature space have not been explored prior to this work. Integrating synthetic instances, which are not direct mappings of any raw input sample, effectively into a supervised learning framework is a non-trivial challenge. Our main idea is to use synthetic instances as the supervising targets in the deep feature space, to implement a representation learning which induce better class distinction in the acquired space.

The proposed framework, *Deep Over-sampling* (DOS), employs a basic CNN architecture in which the lower layers acquire the embedding function and the top layers acquire the classification function. We implement the training of the CNN with explicit supervising information for both functions, i.e., the network parameters are updated by propagation from the output of the lower layers as well as the top layers. Accordingly, the training data presents with each raw input sample, a class label and a target in the deep feature space. The targets are sampled from a linear subspace of the in-class neighbors around the embedded input. As such targets naturally distribute closer to the class mean, our aim is to induce smaller in-class variance among the embeddings.

DOS provides the framework to address the effect of class imbalance on both classifier and representation learning. First, the training data is augmented by assigning multiple synthetic targets to one input sample. Secondly, an iterative

process of learning the CNN and updating the targets with the acquired representation enhances the discriminative power of the deep features.

The main contribution of this work is a general re-sampling framework, which enables the deep neural net to learn the deep representation and the classifier jointly in a class-imbalanced setting without substantial modification on its architecture, thus are applicable to a wide range of deep learning models. We validate the effectiveness of the proposed framework in present an empirical study using public image classification benchmarks. Furthermore, we investigate the effect of the proposed framework outside the class imbalance setting. The rest of this paper is organized as follows. Section 2 introduces the related work and the preliminaries, respectively. Section 3 describes the details of the proposed framework. The empirical results are shown in Section 4, and we present our conclusion in Section 5.

2 Background

2.1 Class Imbalance

Class imbalance is a common issue in practical classification problems, where a large imbalance in the number of training samples between classes causes the learning algorithms to over-generalize for the classes in the majority. Its effect is critical as retrieving the minority classes is usually the primary interest in practice [11, 17]. The countermeasures against class imbalance can be generally categorized into three major approaches. The re-sampling approach attempts to directly adjust the sample sizes by over- or under-sampling on the training set. The instance weighting approach exploits a similar intuition by increasing the importance of the minority class samples. Finally, the cost-sensitive learning approach modifies the loss function and/or the learning algorithm to penalize the errors on the minority class predictions.

For traditional classification models, the synthetic over-sampling methods such as SMOTE [3] have been generally successful in countering the effect of imbalance. Typically, synthetic samples are generated by randomly selecting a minority class sample and taking an interpolation between its neighbors. The re-sampling approach has also been attempted on neural network models, e.g., [28] has combined over-sampling with cost-sensitive learning and [15] has combined under-sampling with synthetic over-sampling.

One limitation of the synthetic over-sampling method is the need for the vector form input data, i.e., it is not applicable to non-vector input domain, e.g., pair-wise distances [16, 1]. Moreover, it implicitly assumes that the interpolations do not deviate from the original distribution, as in the locally linear feature space. Such assumption is usually not problematic for the traditional classification models, many of which are developed with similar assumptions. Synthetic over-sampling is generally successful when the features are pre-selected for such models. Meanwhile, the assumption does not hold for complex, structured data often handled by deep neural nets. Generating samples of complex data is substantially difficult, and simple interpolations can easily deviate from the original

distribution. While acquiring locally-linear representation is a key advantage of deep neural nets, a recent study has reported that class imbalance can affect their representation learning capability as well [14].

In [14], a sophisticated under-sampling scheme called Large Margin Local Embedding (LMLE) was implemented to generate an abridged training set for representation learning. These samples were selected considering the class and cluster structures such as in-/out-of-class and, in-/out-of-cluster neighbors. It also introduced a new loss function based on class-separating margins, inspired by the Large Margin Nearest Neighbor (LMNN) [26].

The potential demerit of under-sampling is the loss of information from discarding the subset of the training data. As such, computationally-intensive analysis to retain important samples, in this case the analyses of class and cluster structure, is inevitable in the re-sampling process. Another drawback in the above work was the specificity of the loss function and the network architecture. The effect of class imbalance, as we demonstrate in the next section, differ based on the classification model. It is thus not clear whether the experimental results of a modified kNN extends generally to other classifiers, especially given that the proposed margin-based loss function is oriented toward the kNN-based classification model. Additionally, its architecture does not support simultaneous learning of the representation and the classifier, which is a key feature of CNN. Overall, the above implementation is likely to require much task- and model-specific configurations when applying to a new problem.

In this paper, we explore an over-sampling scheme in the deep representation space to avoid computationally intensive analyses. We also attempt to utilize a basic CNN architecture in order to maintain its wide applicability and its advantage of cohesive representation and classifier learning.

2.2 Preliminary Results

To motivate our study, we first present a preliminary result on the effect of class imbalance on deep learning. An artificial imbalanced setting was created with the MNIST-back-rotation images [20] by selecting four digits randomly, and removing 90 % of their samples. We trained two instances of a basic CNN architecture [22] by back-propagation respectively with the original and the imbalanced data.

The training of the two CNNs from initial parameters were repeated ten times and the averages of their class-wise retrieval performances, i.e., Precision, Recall, and F1-score, are reported here. Although the overall accuracy has been reported in prior studies, we preferred the class-wise retrieval measures in order to obtain separate insights for the minority and majority classes.

Tables 1a and 1b show the class-wise precision, recall, and F1-score of one trial from each experiment. In Table 1b, the minority class digits are indicated by the asterisks. Additionally, significant declines (0.1 or more) in precision or recall, compared to Table 1a, are indicated by double underline and smaller (0.05 or more) drops are indicated by single underlines. Tables 2a and 2b show the same performance measures by the kNN classifier using the deep representation acquired by the two CNNs. The performance of the kNN, which is a non-inductive

Table 1: Class-wise Performance Comparison (CNN)

(a) Balanced Data

Digit	Precision	Recall	F1-score
0	0.88	0.92	0.90
1	0.93	0.87	0.90
2	0.63	0.68	0.65
3	0.80	0.81	0.80
4	0.64	0.83	0.72
5	0.71	0.69	0.70
6	0.77	0.67	0.72
7	0.75	0.72	0.74
8	0.78	0.73	0.75
9	0.71	0.70	0.70

(b) Imbalanced Data

Digit	Precision	Recall	F1-score
0*	0.60	0.98	0.75
1	0.92	0.78	0.85
2	0.69	0.42	0.52
3	0.81	0.58	0.68
4*	0.090	0.92	0.16
5*	0.12	0.90	0.22
6*	0.075	0.84	0.14
7	0.77	0.53	0.63
8	0.70	0.62	0.65
9	0.80	0.38	0.51

Table 2: Class-wise Performance Comparison (Deep Representation + kNN)

(a) Balanced Data

Digit	Precision	Recall	F1-score
0	0.91	0.86	0.89
1	0.93	0.82	0.87
2	0.60	0.74	0.66
3	0.81	0.74	0.77
4	0.72	0.72	0.72
5	0.64	0.73	0.68
6	0.67	0.78	0.72
7	0.74	0.71	0.72
8	0.74	0.75	0.74
9	0.70	0.64	0.67

(b) Imbalanced Data

Digit	Precision	Recall	F1-score
0*	0.86	0.83	0.85
1	0.91	0.83	0.87
2	0.57	0.62	0.59
3	0.75	0.74	0.74
4*	0.61	0.61	0.61
5*	0.51	0.57	0.54
6*	0.48	0.56	0.52
7	0.69	0.67	0.68
8	0.71	0.67	0.69
9	0.63	0.59	0.61

lazy learning algorithm, is said to substantially reflect the effect of class imbalance on representation learning [14]. The minority classes and reductions in precision or recall are indicated in the same manner as in Table 2a.

In Table 1b, there is a clear trend of decrease in precision for the minority classes. Over the majority classes, there are reduction in recall which are smaller but still substantial. In Table 2b, both the precision and the recall decline for most of the minority classes. There are declines in precision or recall of many majority classes as well.

Table 3 shows the average measures of minority and majority classes over ten trials. The digits of the minority classes were chosen randomly in each trial. The trends in Table 3 regarding the precision and the recall are consistent with those of Tables 1 and 2. These preliminary results support our insight that the class imbalance has a negative impact on the representation learning of CNN as well as its classifier training, and the influence differs depending on the classifier.

3 Deep Over-sampling Framework

This section describes the details of the proposed framework, *Deep Over-sampling* (DOS). The main idea of DOS is to re-sample the training data in an expressive,

Table 3: Summary of Average Retrieval Measures

(a) CNN

Setting	Precision	Recall	F1-score
Balanced	0.76	0.76	0.76
Minority	0.32	0.86	0.43
Majority	0.79	0.57	0.66

(b) Deep Representation + kNN

Setting	Precision	Recall	F1-score
Balanced	0.75	0.75	0.75
Minority	0.65	0.67	0.65
Majority	0.71	0.70	0.71

nonlinear feature space acquired by the convolutional neural network. While previous over-sampling based methods such as SMOTE have achieved general success for traditional models, their approach of sampling from the linear subspace of the original data has clear limitations for complex, structured data, such as imagery. In contrast, DOS implements re-sampling in the linear subspace of deep feature instances and exploit the re-sampled instances for explicitly supervised representation learning as well as complementing the minority classes.

3.1 Notations

We employ a basic CNN whose architecture can be divided into two groups of layers: the lower layers embedding the raw input into the deep feature space, and the top layers predicting the class label from the deep features. We denote the embedding function of the CNN by $f : \Phi \rightarrow \mathbb{R}^d$, where Φ is the raw input domain with a complex data structure. We also denote the discriminative function of the CNN by $g : \mathbb{R}^d \rightarrow [0 : 1]^n$, whose output represents a vector of posterior class probabilities $P(C|x)$ over n classes.

Let $\mathcal{X} = \{(x_i, y_i)\}_{i=1}^m$ denote a set of training data, where $x_i \in \Phi$ and y_i takes a class value from $\mathcal{C} = \{c_j\}_{j=1}^n$. The network parameters, denoted by $\mathbf{W}_f$ and $\mathbf{W}_g$ for the embedding layers and the classification layers, respectively, are learned with back-propagation. A class imbalance, such that $\#\{(x, y) : y = c_i\} \gg \#\{(x, y) : y = c_j\}$ for some $\{c_i, c_j\} \subset \mathcal{C}$, may arise from practical issues such as the cost of data collection. A significant imbalance can hinder the performance of the acquired model. The architecture is illustrated in Fig.1. We further elaborate on its details in Section 3.3.

3.2 Deep Feature Overloading

We employ re-sampling in the deep feature space to assign each raw input sample with multiple deep feature instances. As a result, the supervising targets are provided for both the embedding function f and the classification function g.

Let $\mathcal{V}(c_j) = \{f(x_i) : y_i = c_j\}$ denote the set of embeddings whose raw input has the label c_j. A training instance is defined as a triplet $z_i = (x_i, y_i, \mathcal{N}(x_i))$, consisting of an input sample x_i, its class label y_i, and a subset of embeddings

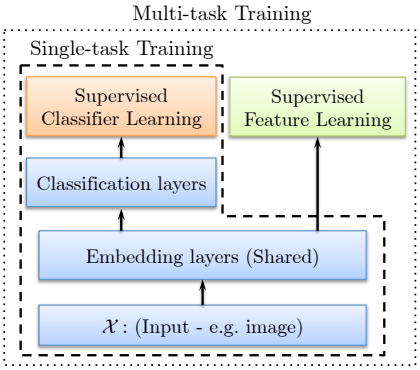

Fig. 1: CNN Architecture

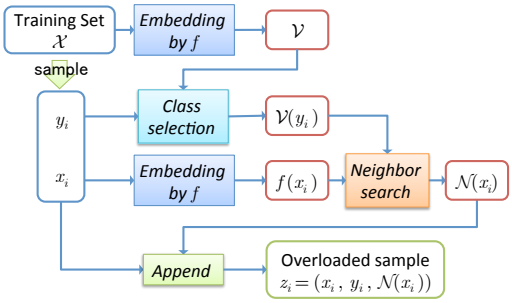

Fig. 2: Deep Feature Overloading

$\mathcal{N}(x_i)$. $\mathcal{N}(x_i)$ is a subset of $\mathcal{V}(y_i)$ that includes $f(x_i)$ and its k in-class neighbors,

$$\mathcal{N}(x_i; k) = \underset{\substack{\mathcal{N} \subset \mathcal{V}(y_i) \wedge \\ \#(\mathcal{N}) = k+1}}{\arg\min} \sum_{v \in \mathcal{N}} \|f(x_i) - v\|^2 \tag{1}$$

We refer to the process of pairing each $(x_i, y_i) \in \mathcal{X}$ with its deep feature neighbors as deep feature *overloading*. The process is illustrated in Fig.2. We refer to k as the *overloading* parameter and $\mathcal{Z} = \{z_i\}_{i=1}^m$ as the *overloaded* training set.

As we describe in the following section, the deep feature neighbors are used to generate synthetic targets for the minority class samples. The value of k, thus may be varied between the minority and the majority classes as the latter does not need to be synthetically complemented. Note that the minimum value for k is 0, in which case $\mathcal{N}(x_i) = \{f(x_i)\}$.

3.3 Micro-cluster Loss Function

Our CNN architecture, illustrated in Fig.1, features two outputs, one for the classification and one for the embedding functions. The initial parameters of the network are learned in a single-task training using only the substructure indicated by the dotted box on the left side of the figure with the original imblanced training set.

The classifier output is given by the softmax layer and the cross-entropy loss [9] $\mathcal{H}$ based on the predicted class probabilities $g(f(x))$ of a given input (x, y)

$$\ell(x, y) = \mathcal{H}\left(g(f(x)), y\right) \tag{2}$$

is used for single-task learning.

After the initialization, the training is expanded to the multi-task learning architecture in Fig.1 to use propagation from both f and g. The loss function given an overloaded sample z_i is defined with regards to $\mathcal{N}(x_i)$, which can be

considered an in-class cluster in the deep feature space. We thus refer to the functions as the *micro-cluster* loss.

The micro-cluster loss for the embedding function f is defined as a sum of squared errors

$$\ell_f(x) = \sum_{v \in \mathcal{N}(x)} \|f(x) - v\|^2 \tag{3}$$

The minimum of (3) is obtained when $f(x)$ is mapped to the mean of $\mathcal{N}(x_i)$. Note that the mean is a synthetic point in the deep feature space, to which no particular original sample is projected.

There are two key intuitions for setting the target representation to local means. First, the summation of the squared errors can add emphases to the minority class samples, by *overloading* them with a larger number of embeddings. Secondly, as the local means distribute closer toward the mean of the original distribution, it induces smaller in-class variance in the learned representation. Smaller in-class variance yields better class distinction, which can be induced further by iterating the procedure with the updated embeddings.

The micro-cluster loss for g is defined as the weighted sum of the cross-entropy losses, i.e.,

$$\ell_g(x, y) = \sum_{v \in \mathcal{N}(x)} \rho(v) \mathcal{H}(g(v), y) \tag{4}$$

where $\rho(v)$ is the normalized exponential weight given the squared errors in (3),

$$\rho(v) = \frac{1}{Z} \exp\left(-\|f(x) - v\|^2\right) \tag{5}$$

and Z denotes a normalizer such that

$$Z = \sum_{\mathcal{N}(x)} \exp\left(-\|f(x) - v\|^2\right)$$

In (5), the largest weight among $\mathcal{N}(x)$, 1, is assigned to the original loss from $f(x)$, and a larger weight is assigned to neighbors in a closer range.

3.4 Deep Over-sampling

Deep Over-sampling uses the overloaded instances to supplement the minority classes. Multiple overloaded instances are generated from each training sample by pairing it with different targets sampled from the linear subspace of its in-class neighbors.

Let $\mathcal{W}$ denote a domain of positive, $\ell 1$-normalized vectors of k-dimensions, i.e., for all $\mathbf{w} \in \mathcal{W}$, $\|\mathbf{w}\|^1 = 1$ and $w_i \geq 0$ for $i = 1, \ldots, k$. Note that k is the overloading parameter. For each overloaded instance $z_i \in \mathcal{Z}$, we sample a set of vectors $\{\mathbf{w}^{(i,j)}\}_{j=1}^{r}$ from $\mathcal{W}$.

We define the weighted overloading instance as a quadruplet

$$z_i^{(j)} = (x_i, y_i, \mathcal{N}(x_i), \mathbf{w}^{(i,j)}) \tag{6}$$

Note that each element of the weight vector correspond with an element of $\mathcal{N}(x_i)$.

Sampling r vectors for each z_i, we obtain a weighted training set

$$\mathcal{Z}' = \bigcup_{j=1}^{r} \left\{ z_i^{(j)} \right\}_{i=1}^{m} \tag{7}$$

We define the following micro-cluster loss functions for the weighted instances. The loss function for f given a quadruplet of x, y, $\mathcal{N}(x) = \{v_i\}_{i=1}^{k}$), and $\mathbf{w} = (w_1, \ldots, w_k)$ is written as

$$\ell'_f(x, y, \mathbf{w}, \mathcal{N}(x)) = \sum_{i=1}^{k} w_i \left\| f(x) - v_i \right\|^2 \tag{8}$$

The minimum of (8) is attained when $f(x)$ is at the weighted mean, $\sum_i w_i v_i$.

The weighted micro-cluster loss for g is defined, similarly to (4), as

$$\ell'_g(x, y, \mathbf{w}) = \sum_{i=1}^{k} \rho'(v_i, w_i) \mathcal{H}(g(v_i), y) \tag{9}$$

where ρ' is the normalized weight

$$\rho'(v_i, w_i) = \frac{1}{Z} \exp(-w_i \| f(x) - v_i \|^2) \tag{10}$$

To summarize, the augmentative samples for the minority classes are generated by pairing each raw input sample with multiple targets for representation learning. The rationale for learning to map one input onto multiple targets can be explained as promoting robustness under the imbalanced setting. Since there are less than sufficient number of samples for the minority classes, strong supervised learning may induce the risk of overfitting. Generating multiple targets within the range of local neighbors is similar in effect as adding noise to the target and can prevent the convergence of the gradient descent to undesired local solutions.

After training the CNN, the targets are recomputed with the updated representation. The iterative process of training the CNN and updating the targets incrementally shifts the targets toward the class mean and improve the class distinction among the the embeddings. The pseudo code of the iterative procedure is shown in Algorithm 1.

3.5 Parameter Selection

As mentioned in Section 3.2, different values of overloading parameter k may be selected for the minority and the majority classes to placing additional emphases on the former. Let k_{mnr} and k_{mjr} denote the overloading values for the minority and the majority classes, respectively. If the latter is set to the minimum, i.e., $k_{\text{mjr}} = 0$, then the loss for the minority class samples, as given by (3), accounts for $(k_{\text{mnr}} + 1)$ times more squared errors.

Algorithm 1 CNN Training with Deep Over-sampling

Input $\mathcal{X} = \{(x_i, y_i)\}_{i=1}^m$, class values $\mathcal{C} = \{c_j\}_{j=1}^n$, class-wise overloading $\{k_j\}_{j=1}^n$, class-wise over-sampling size $\{r_j\}_{j=1}^n$, CNN with outputs for functions f and g, deep feature dimensions d, number of iterations T
Output A trained CNN with functions f and g
function STL($\mathcal{T}$): Single-task training of CNN with training set $\mathcal{T}$
function MTL($\mathcal{T}$): Multi-task training of CNN with training set $\mathcal{T}$
Initialize CNN parameters: STL($\mathcal{X}$)
for $t = 1, \ldots, T$ **do**
  for $i = 1, \ldots, m$ **do**
    Compute $v_i = f(x_i)$
  end for
  for $j = 1, \ldots, n$ **do**
    if $k_j > 0$ **then**
      Compute the mutual distance matrix $D(c_j)$ over $\mathcal{V}(c_j)$ for neighbor search
    end if
    $\mathcal{Z}_j = \emptyset$
    for $\{(x_i, y_i) : y_i = c_j\}$ **do**
      Select $\mathcal{N}(x_i)$ from $\mathcal{V}(c_j)$
      Sample a set of r_j normalized positive vectors $\mathcal{W}$
      $\mathcal{Z}_j = \mathcal{Z}_j \cup \{(x_i, y_i, \mathcal{N}(x_i), \mathbf{w})\}_{\mathbf{w} \in \mathcal{W}}$
    end for
  end for
  $\mathcal{Z} = \bigcup_{j=1}^n \mathcal{Z}_c$
  Update CNN parameters: MTL($\mathcal{Z}$)
end for

In essence, however, k should be chosen to reflect the extent of the neighborhood, as the size of the neighbors, $\mathcal{N}(x)$, can influence the the efficiency of the back-propagation learning. As one increase k, the target shifts closer to the global class mean and, in turn, farther away from the tentative embedding $f(x)$. For better convergence of the gradient descent, the target should be maintained within a moderate range of proximity from the tentative embedding.

Our general guideline for the parameter selection is therefore to choose a value of k_{mnr} from [3:10] by empirical validation and set $k_{\text{mjr}} = 0$ provided that there are sufficient number of samples for the majority classes. Furthermore, we suggest to choose the over-sampling rate r from $[\frac{1}{R} : \frac{k_{\text{mnr}}}{R}]$ where R denotes the average ratio of the minority and the majority class samples, i.e.,

$$R = \frac{\#\{(x, y) : (x, y) \in \mathcal{X} \wedge y = c_{\text{mnr}}\}}{\#\{(x, y) : (x, y) \in \mathcal{X} \wedge y = c_{\text{mjr}}\}} \tag{11}$$

For the number of iterations T, we suggest it to be the same as the number of training rounds, i.e., to re-compute the targets after every training round.

4 Empirical Results

We conducted an empirical study[3] to evaluate the DOS framework in three experimental settings. The first experiment is a baseline comparison, for which we replicated a setting used in the most recently proposed model to address the class imbalance. Secondly, we evaluated the sensitivity of DOS with different levels of imbalance and parameter choices. Finally, we investigated the effect of deep over-sampling in the standard, balanced settings. The imbalanced settings were created with standard benchmarks by deleting the samples from selected classes.

4.1 Datasets and CNN Settings

We present the results on five public datasets: MNIST [21], MNIST-*back-rotation* images [20], SVHN [23], CIFAR-10 [18], and STL-10 [6]. We have set up the experiment in the image domain, because it is one of the most popular domains in which CNNs have been used extensively, and also it is difficult to apply the SMOTE algorithm directly. Note that we omit the result of preprocessing the imbalanced image set using SMOTE, as it achieved no improvement in the classifier performances.

The MNIST digit dataset consists of a training set of 60000 images and a test set of 10000 images, which includes 6000 and 1000 images for each digit, respectively. The MNIST-*back-rotation*-image (MNIST*rb*) is an extension of the MNIST dataset contains 28×28 images of rotated digits over randomly inserted backgrounds. The default training and test set consist of 12000 and 50000 images, respectively. The Street View House Numbers (SVHN) dataset consists of 73,257 digits for training and 26,032 digits for testing in 32×32 RGB images. The CIFAR-10 dataset consists of 32×32 RGB images. A total of 60,000 images in 10 categories are split into 50,000 training and 10,000 testing images. The STL-10 dataset contains 96×96 RGB images in 10 categories. All results are reported on the default test sets.

For MNIST, MNIST*rb*, and SVHN, we employ a CNN architecture consisting of two convolution layers with 6 and 16 filters, respectively, and two fully-connected layers with 400 and 120 hidden units. ReLU is adopted between the convolutional layers and the fully connected layers. For CIFAR-10 and STL-10, we use convolutional layers with 20 and 50 filters and fully-connected layers with 500 and 120 hidden units. The summary of the datasets and the architectures are shown in Table 4.

4.2 Experimental Settings and Evaluation Metrics

Our first experiment follows that of [14] using the MNIST-*back-rot* images. First, the original dataset was augmented 10 times with mirrored and rotated images.

[3] The source codes for reproducing the datasets and the results are made available at http://www.rs.tus.ac.jp/ando/exp/DOS.html

Table 4: Datasets and CNN Architectures

Dataset	$\#\mathcal{C}$	#TRN	#TST	Image Dim	CNN Layers	Batch Size/ Trn.Rnds
MNIST [21]	10	50,000	10,000	$1 \times 28 \times 28$	C6-C16-F400-F120	60/3
MNISTrb [20]	10	12,000	50,000	$1 \times 28 \times 28$	C6-C16-F400-F120	40/5
SVHN [23]	10	73,257	26,032	$3 \times 32 \times 32$	C6-C16-F400-F120	60/3
CIFAR-10 [18]	10	50,000	10,000	$3 \times 32 \times 32$	C20-C50-F500-F120	50/4
STL-10 [6]	10	5,000	8,000	$3 \times 96 \times 96$	C20-C50-F500-F120	50/5

Then, class imbalance was created by deleting samples selected with Gaussian distribution until a designated overall reduction rate is reached. We compare the average per-class accuracy (average class-wise recall) with those of Triplet re-sampling with cost-sensitive learning and Large Margin Local Embedding (LMLE) reported in [14]. Triplet loss re-sampling with cost-sensitive learning is a hybrid method that implements the triplet loss function used in [5, 24, 25] with re-sampling and cost-sensitive learning.

The second experiment analyzes the sensitivity of DOS over the levels of imbalance and the choices of k, using MNIST, MNIST-back-rotation, and SVHN datasets. The value of k is altered over 3, 5, and 10. In [3], 5 was given as the default value of k and other values have been tested in ensuing studies. The imbalance is created by randomly selecting four classes and removing p portion of their samples. We report the class-wise retrieval measures: precision, recall, F1-score, and the Area Under the Precision-Recall Curve (AUPRC), for the minority and the majority classes, respectively. The precision-recall curve is used in similar scope as the receiver operating characteristic curve (ROC). [10] has suggested the use of AUPRC over AUROC, which provides an overly optimistic estimate of the retrieval performance in some cases. Note that the precision, recall, and F1-score are computed from a multi-class confusion matrix, while the precision-recall curve is computed from the class-wise posterior probabilities.

The third experiment is conducted using the original SVHN, CIFAR-10, and STL-10 datasets. Since the classes are not imbalanced, the overloading value k is set uniformly for all classes, and the over-sampling rate r, from (11), is set to 1. The result of this experiment thus reflect the effect of deep feature overloading in a well-balanced setting. The evaluation metrics are the same as the second experiment, but averaged over all classes.

4.3 Results

Comparison with Existing work The result from the first experiment is summarized in Table 5. The overall reduction rate is shown on the first column. The performances of the baseline methods (TL-RS-CSL, LMLE) are shown in the second and the third columns. In the last three columns, the performances of DOS and two classifiers: logistic regression (LR) and k-nearest neighbors (kNN) using its deep representation are shown. While all methods show declining trends of accuracy, DOS showed the slowest decline against the reduction rate.

Table 5: Baseline Comparison (Class-wise Recall)

Reduction Rate	TL-RS-CSL	LMLE	DOS	DOS-LR	DOS-kNN
0	76.12	77.64	77.35	77.62	76.00
20	67.18	75.58	77.13	74.60	73.53
40	56.49	70.13	75.43	73.53	72.98

Sensitivity Analysis on Imbalanced Data Tables 6 and 7 summarize the results of the second experiment. In Table 6, we compare the performances of the basic CNN, traditional classifiers using the deep representation of the basic CNN (CNN-CL), and DOS, over the reduction rates $p = 0.90, 0.95, 0.99$. On each row, four evaluation measures on MNIST, MNISTbr, and SVHN are shown. Note that the AUPRC of CNN-CL is computed from the predicted class probabilities of the logistic regression classifier and other performances are those of the kNN classifier. The performances of the minority (mnr) and the majority (mjr) classes are indicated on the third column, respectively. We indicate the significant increases (0.1 or more) over CNN and CNN-CL by DOS with double underlines and smaller increases (0.05 or more) with single underlines. In Table 6, DOS exhibit more significant advantage with increasing level of imbalance, over the basic CNN and the classifiers using its deep representation.

Table 7 summarizes the sensitivity analysis on the overloading parameter values $k = 3, 5, 10$ with reduction rate set to $p = 0.01$. Each row shows the four evaluation measures on SVHN, CIFAR-10, and STL-10, respectively. For reference, The performances of the basic CNN and the deep feature classifiers are shown on the top rows. We indicate the significant increases over the baselines in the similar manner as Table 6. The minority and majority classes are indicated on the third column as well. Additionally, we indicate the unique largest values among DOS settings by bold letters. These results show that DOS is generally not sensitive to the choice of k. However, there is a marginal trend that the performances on the minority classes are slightly higher with $k = 3, 5$ and those of the majority classes are slightly higher with $k = 10$. It suggests possible decline in performance with overly large k.

Run-time analysis The deep learning in the above experiment was conducted on NVIDIA GTX 980 graphics card with 704 cores and 6GB of global memory. The average increases in run-time for DOS compared to the basic, single-task learning architecture CNN were 11%, 12%, and 32% for MNIST, MNIST-bakrot, and SVHN datasets, respectively.

Evaluation on Balanced Data Table 8 summarizes the performances on the balanced settings. On the top rows, the performances of the basic CNN and the classifiers using its representations are shown for reference. On the bottom rows, the performances of DOS at $k = 3, 5, 10$ are shown. The uniquely best values among the three settings are indicated by bold fonts. While the improvements

Table 6: Performance Comparison on Imbalanced Data over Reduction Rate

model	p	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	1.0	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	1.0	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	0.039	0.68	0.069	0.61
		mjr	0.89	0.99	0.94	1.0	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	1.0	0.66	0.77	0.71	0.79	0.71	0.82	0.72	0.84
		mjr	0.98	0.99	0.98	1.0	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	1.0	0.56	0.75	0.63	0.72	0.40	0.89	0.55	0.73
		mjr	0.97	0.99	0.98	1.0	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	1.0	0.77	0.70	0.73	0.82	0.89	0.68	0.77	0.90

Table 7: Performance Comparison on Imbalanced Data over k

Classifier	k		MNIST				MNIST-back-rot				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	1.0	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	1.0	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	1.0	**0.77**	**0.70**	**0.73**	**0.82**	**0.90**	0.69	0.78	**0.91**

Table 8: Performance Comparison on Balanced Data over k

Classifier	k	SVHN		CIFAR-10		STL-10	
		F1	AUC	F1	AUC	F1	AUC
CNN		0.85	0.92	0.62	0.68	0.38	0.38
CNN-CL		0.85	0.87	0.61	0.68	0.39	0.40
DOS	3	0.87	0.94	0.64	0.70	0.42	0.41
	5	0.88	**0.95**	0.64	**0.71**	0.42	0.42
	10	0.88	0.94	0.64	0.70	0.42	0.42

from the basic CNN were smaller (between 0.01 and 0.03) than in previous experiments, DOS showed consistent improvements across all datasets. This result supports our view that deep feature overloading can improve the discriminative power of the deep representation. We note that the performance of DOS were not sensitive to the values chosen for k.

5 Conclusion

We proposed the Deep Over-sampling framework for imbalanced classification problem of complex, structured data that allows the CNN to learn the deep representation and the classifier jointly without substantial modification to its architecture. The framework extends the synthetic over-sampling technique by using the synthetic instances not only to complement the minority classes for classifier learning, but also to supervise representation learning and enhance its robustness and class distinction. The empirical results showed that the proposed framework can address the class imbalance more effectively than the existing countermeasures for deep learning, and the improvements were more significant under stronger levels of imbalance. Furthermore, its merit on representation learning were verified from the improved performances in the balanced setting.

References

1. Ando, S.: Classifying imbalanced data in distance-based feature space. Knowledge and Information Systems 46(3), 707–730 (2016)
2. Bengio, Y., Courville, A., Vincent, P.: Representation learning: A review and new perspectives. IEEE Trans. Pattern Anal. Mach. Intell. 35(8), 1798–1828 (2013)
3. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic Minority Over-sampling Technique. J. Artif. Int. Res. 16(1), 321–357 (2002)
4. Chawla, N.V., Cieslak, D.A., Hall, L.O., Joshi, A.: Automatically Countering Imbalance and Its Empirical Relationship to Costs. Data Min. Knowl. Discov. 17(2), 225–252 (2008)
5. Chechik, G., Shalit, U., Sharma, V., Bengio, S.: An online algorithm for large scale image similarity learning. In: Advances in Neural Information Processing Systems 22, pp. 306–314. (2009)
6. Coates, A., Lee, H., Ng, A.: An analysis of single-layer networks in unsupervised feature learning. In: Procs. of the 14th Intl. Conf. on Artificial Intelligence and Statistics. vol. 15, pp. 215–223. (2011)
7. Collobert, R., Weston, J.: A unified architecture for natural language processing: Deep neural networks with multitask learning. In: Procs. of the 25th Intl. Conf. on Machine Learning. pp. 160–167. (2008)
8. Dong, C., Loy, C.C., He, K., Tang, X.: Learning a Deep Convolutional Network for Image Super-Resolution, pp. 184–199. Springer International (2014)
9. Dunne, R.A.: A Statistical Approach to Neural Networks for Pattern Recognition (Wiley Series in Computational Statistics). Wiley-Interscience (2007)
10. Flach, P.A., Hernández-Orallo, J., Ramirez, C.F.: A coherent interpretation of auc as a measure of aggregated classification performance. In: Procs. of the 28th Intl. Conf. on Machine Learning, 2011. pp. 657–664. (2011)

11. He, H., Garcia, E.A.: Learning from Imbalanced Data. IEEE Trans. on Knowl. and Data Eng. 21(9), 1263–1284 (2009)
12. Hinton, G., Deng, L., Yu, D., Dahl, G.E., r. Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., Kingsbury, B.: Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal Processing Magazine 29(6), 82–97 (2012)
13. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science 313(5786), 504–507 (2006)
14. Huang, C., Li, Y., Loy, C.C., Tang, X.: Learning deep representation for imbalanced classification. In: 2016 IEEE Conf. on Computer Vision and Pattern Recognition. pp. 5375–5384 (2016)
15. Jeatrakul, P., Wong, K.W., Fung, C.C.: Classification of imbalanced data by combining the complementary neural network and smote algorithm. In: Procs. of the 17th Intl. Conf. on Neural Information Processing: Models and Applications - Volume Part II. pp. 152–159. ICONIP'10, (2010)
16. Köknar-Tezel, S., Latecki, L.J.: Improving SVM Classification on Imbalanced Time Series Data Sets with Ghost Points. Knowl. Inf. Syst. 28(1), 1–23 (2011)
17. Krawczyk, B.: Learning from imbalanced data: open challenges and future directions. Progress in Artificial Intelligence 5(4), 221–232 (2016)
18. Krizhevsky, A.: Learning Multiple Layers of Features from Tiny Images. Master's thesis (2009)
19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Procs. of the 25th Intl. Conf. on Neural Information Processing Systems. pp. 1097–1105. (2012)
20. Larochelle, H., Erhan, D., Courville, A., Bergstra, J., Bengio, Y.: An empirical evaluation of deep architectures on problems with many factors of variation. In: Procs. of the 24th Intl. Conf. on Machine Learning. pp. 473–480. (2007)
21. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Procs. of the IEEE 86(11), 2278–2324 (1998)
22. LeCun, Y., Kavukcuoglu, K., Farabet, C.: Convolutional networks and applications in vision. In: Procs. of 2010 IEEE Intl. Symposium on Circuits and Systems. pp. 253–256 (2010)
23. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning. In: NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011 (2011)
24. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: 2015 IEEE Conf. on Computer Vision and Pattern Recognition. pp. 815–823 (2015)
25. Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., Wu, Y.: Learning fine-grained image similarity with deep ranking. In: Procs. of the 2014 IEEE Conf. on Computer Vision and Pattern Recognition. pp. 1386–1393. (2014)
26. Weinberger, K.Q., Saul, L.K.: Distance Metric Learning for Large Margin Nearest Neighbor Classification. J. Mach. Learn. Res. 10, 207–244 (2009)
27. Zeiler, M.D., Fergus, R.: Computer Vision, ECCV 2014: 13th European Conf., Zurich, Procs., vol. 1, chap. Visualizing and Understanding Convolutional Networks, pp. 818–833. (2014)
28. Zhou, Z.H., Liu, X.Y.: Training cost-sensitive neural networks with methods addressing the class imbalance problem. IEEE Trans. on Knowl. and Data Eng. 18(1), 63–77 (2006)