Expert Systems With Applications 91 (2018) 464–471

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Effective data generation for imbalanced learning using conditional generative adversarial networks

Georgios Douzas, Fernando Bacao*

NOVA Information Management School, Universidade Nova de Lisboa, Portugal

ARTICLE INFO

Article history:
Received 1 June 2017
Revised 17 August 2017
Accepted 11 September 2017
Available online 13 September 2017

Keywords:
GAN
Imbalanced learning
Artificial data
Minority class

ABSTRACT

Learning from imbalanced datasets is a frequent but challenging task for standard classification algorithms. Although there are different strategies to address this problem, methods that generate artificial data for the minority class constitute a more general approach compared to algorithmic modifications. Standard oversampling methods are variations of the SMOTE algorithm, which generates synthetic samples along the line segment that joins minority class samples. Therefore, these approaches are based on local information, rather on the overall minority class distribution. Contrary to these algorithms, in this paper the conditional version of Generative Adversarial Networks (cGAN) is used to approximate the true data distribution and generate data for the minority class of various imbalanced datasets. The performance of cGAN is compared against multiple standard oversampling algorithms. We present empirical results that show a significant improvement in the quality of the generated data when cGAN is used as an oversampling algorithm.

1. Introduction

Learning from imbalanced data is an important problem for the research community as well as the industry practitioners (Chawla, Japkowicz, & Kolcz, 2003). An imbalanced learning problem can be defined as a learning problem from a binary or multiple-class dataset where the number of instances for one of the classes, called the majority class, is significantly higher than the number of instances for the rest of the classes, called the minority classes (Chawla, Bowyer, Hall, & Kegelmeyer, 2002). The Imbalance Ratio (IR), defined as the ratio between the majority class and each of the minority classes, varies for different applications and for binary problems values between 100 and 100.000 have been observed (Chawla et al., 2002; Barua, Islam, Yao, & Murase, 2014).

Imbalanced data are a characteristic of multiple real-world applications such as medical diagnosis, information retrieval systems, fraud detection, detection of oil spills in radar images, direct marketing, automatic classification of land use and land cover in remote sensing images, detection of rare particles in experimental high-energy physics, telecommunications management and bioinformatics (Akbani, Kwek, & Japkowicz, 2004; He & Garcia, 2009; Clearwater & Stern, 1991; Graves et al., 2016; Verbeke, Dejaeger, Martens, Hur, & Baesens, 2012; Zhao, Li, Chen, & Aihara, 2008). Standard learning methods perform poorly in imbalanced data sets as they induce a bias in favor of the majority class. Specifically, during the training of a standard classification method the minority classes contribute less to the minimization of the objective function. Also the distinction between noisy and minority class instances is often difficult. An important observation is that in many of these applications the misclassification cost of the minority classes is often higher than the misclassification cost of the majority class (Domingos, 1999; Ting, 2002). Therefore the methods that address the class imbalance problem aim to increase the classification accuracy for the minority classes.

There are three main approaches to deal with the class imbalanced problem (Fernández, López, Galar, Jesus, & Herrera, 2013). The first is the modification/creation of algorithms that reinforce the learning towards the minority class. The second approach is the application of cost-sensitive methods at the data or algorithmic level in order to minimize higher cost errors. The third and more general approach is the modification at the data level by rebalancing the class distribution through under-sampling, over-sampling or hybrid methods.

Our focus in this paper is oversampling techniques, which result in the generation of artificial data for the minority class. Standard oversampling methods are inspired by Synthetic Minority Oversampling Technique (SMOTE) algorithm (Chawla et al., 2002), generating synthetic samples along the line segment that joins minority class samples. A direct approach to the data generation process

* Corresponding author.
E-mail addresses: gdouzas@icloud.com (G. Douzas), bacao@novaims.unl.pt (F. Bacao).

http://dx.doi.org/10.1016/j.eswa.2017.09.030
0957-4174/

would be the use of a generative model that captures the actual data distribution. Generative Adversarial Networks (GAN) is a recent method that uses neural networks to create generative models (Goodfellow et al., 2014). A conditional Generative Adversarial Network (cGAN) extends the GAN model by conditioning the training procedure on external information (Mirza & Osindero, 2014). In this paper we apply a cGAN on binary class imbalanced datasets, where the cGAN conditioning on external information are the class labels of the imbalanced datasets. The final generative model is used to create artificial data for the minority class i.e. the generator corresponds to an oversampling algorithm.

For the evaluation of cGAN as an oversampling method an experimental analysis is performed, based on 12 publicly available datasets from Machine Learning Repository. In order to test it on a wide range of IRs, additional datasets are created by undersampling the minority class of these 12 datasets as well as by adding simulated datasets with appropriate characteristics. Then the proposed method is compared to Random Oversampling, SMOTE algorithm, Borderline SMOTE (Han, Wang, & Mao, 2005), ADASYN (He, Bai, Garcia, & Li, 2008) and Cluster-SMOTE (Cieslak, Chawla, & Striegel, 2006). For the classification of the binary class data five classifiers and three evaluation metrics are applied.

The sections in the paper are organized as follows. In Section 2, an overview of related previous works and existing sampling methods is given. In Section 3, the theory behind GANs is described. Section 4 presents the proposed method in detail. Section 5 presents the research methodology. In Section 6 the experimental results are presented while conclusions are provided in Section 7.

2. Related work

Considering that our focus is the modification on the data level, and particularly the generation of artificial data, we provide a short review of the oversampling methods. A review of the other methods can be found in Galar, Fernández, Barrenechea, Bustince, and Herrera (2012) and Chawla (2005). Oversampling methods generate synthetic examples for the minority class and add them to the training set. A simple approach, known as Random Oversampling, creates new data by copying random minority class examples. The drawback of this approach is that the exact replication of training examples can lead to overfitting since the classifier is exposed to the same information.

An alternative approach that aims to avoid this problem is SMOTE. Synthetic data are generated along the line segment that joins minority class samples. SMOTE has the disadvantage that, since the separation between majority and minority class clusters is not often clear, noisy samples may be generated (He & Garcia, 2009). To avoid this scenario various modifications of SMOTE have been proposed. SMOTE+Edited Nearest Neighbor (Batista, Prati, & Monard, 2004) combination applies the edited nearest neighbor rule (Wilson, 1972) after the generation of artificial examples through SMOTE to remove any misclassified instances, based on the classification by its three nearest neighbors. Safe-Level SMOTE (Bunkhumpornpat, Sinapiromsaran, & Lursinsap, 2009) modifies the SMOTE algorithm by applying a weight degree, the safe level, in the data generation process. Borderline-SMOTE (Han et al., 2005), MWMOTE (Majority Weighted Minority Oversampling Technique for Imbalanced Data Set Learning) (Barua et al., 2014), ADASYN (He et al., 2008) and its variation KernelADASYN (Tang & He 2015) aim to avoid the generation of noisy samples by identifying the borderline instances of the majority and minority classes that in turn are used to identify the informative minority class samples.

The methods above address the problem of between-class imbalance (Nekooeimehr & Lai-Yuen, 2016). Another type of problem is the within-class imbalance (Nekooeimehr & Lai-Yuen, 2016; Bunkhumpornpat, Sinapiromsaran, & Lursinsap, 2012; Cieslak & Chawla, 2008; Jo & Japkowicz, 2004) i.e. when sparse or dense subclusters of minority or majority instances exist. Clustering based oversampling methods that deal with the between-class imbalance problem have recently been proposed. These methods are initially partitioning the input space and then apply sampling methods in order to adjust the size of the various clusters. Cluster-SMOTE applies the k-means algorithm and then generates artificial data by applying SMOTE in the clusters. Similarly DB-SMOTE (Bunkhumpornpat et al., 2012) uses the DB-SCAN algorithm to discover arbitrarily shaped clusters and generates synthetic instances along a shortest path from each minority class instance to a pseudo-centroid of the cluster. A-SUWO (Nekooeimehr & Lai-Yuen, 2016) creates clusters of the minority class instances with a size, which is determined using cross validation and generates synthetic instances based on a proposed weighting system. SOMO (Douzas & Bacao, 2017) creates a two dimensional representation of the input space and based on it, applies the SMOTE procedure to generate intracluster and intercluster synthetic data that preserve the underlying manifold structure. Other types of oversampling approaches are based on ensemble methods (Wang, Minku, & Yao, 2015; Sun et al., 2015) such as SMOTEBoost (Chawla, Lazarevic, Hall, & Bowyer, 2003), DataBoost-IM (Guo & Viktor, 2004).

3. GAN and cGAN algorithms

In this section, we provide a summary of the GAN and cGAN frameworks following closely the notation in Goodfellow et al. (2014) and Gauthier (2015). The GAN is based on the idea of competition, in which a generator G and a discriminator D are trying to outsmart each other. The objective of the generator is to confuse the discriminator. The objective of the discriminator is to distinguish the instances coming from the generator and the instances coming from the original dataset. If the discriminator is able to identify easily the instances coming from the generator then, relative to its discrimination ability, the generator is producing low quality data. We can look at the GAN setup as a training environment for the generator where the discriminator, while also improving, is providing feedback about the quality of the generated instances, forcing the generator to increase its performance.

More formally, the generative model G, defined as $G\colon\ Z \rightarrow X$ where Z is the noise space of arbitrary dimension d_Z that corresponds to a hyperparameter and X is the data space, aims to capture the data distribution. The discriminative model, defined as $D\colon\ X \rightarrow [0,\ 1]$, estimates the probability that a sample came from the data distribution rather than G. These two models, which are both multilayer perceptrons, compete in a two-player minmax game with value function:

$$\min_G \max_D V(D,\ G) = E_D\ +\ E_G$$

where:

$$E_D = E_{x\sim p_{data}(x)}[logD(x)]$$
$$E_G = E_{z\sim p_z(z)}[\log(1 - D(G(z)))] \quad (1)$$

The $x \in X$ values are sampled from the data distribution $p_{data}(x)$ and the $z \in Z$ values are sampled from the noise distribution $p_z(z)$. The training procedure consists of alternating between k optimizing steps for D and one optimizing step for G by applying SGD. Therefore during training, D is optimized to correctly classify training data and samples generated from G, assigning 1 and 0 respectively. On the other hand the generator is optimized to confuse the discriminator by assigning the label 1 to samples generated from G. The unique solution of this adversarial game corresponds to G recovering the data distribution and D equal to ½ for any input (Goodfellow et al., 2014).