SoftwareX ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

SoftwareX

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

Original software publication

ICF: An algorithm for large scale classification with conic functions

Emre Cimen [a,b,*], Gurkan Ozturk [a,b], Omer Nezih Gerek [a]

[a] Anadolu University, Faculty of Engineering, 26555, Eskisehir, Turkey
[b] Anadolu University, Computational Intelligence and Optimization Laboratory (CIOL), 26555, Eskisehir, Turkey

ARTICLE INFO

Article history:
Received 29 November 2017
Accepted 5 December 2017

Keywords:
Polyhedral conic functions
Mathematical programming
Classification
Machine learning

ABSTRACT

Incremental Conic Functions (ICF) algorithm is developed for solving classification problems based on mathematical programming. This algorithm improves previous version of conic function-based classifier construction in terms of computational speed. Furthermore, the incremental step avoids the a-priori knowledge of number of sub-classes (which is a necessary parameter in the clustering step of this classification algorithm). Test results show that ICF is, on the average almost 3-times faster than previous versions without sacrificing accuracy. Python 2.7 implementation and software explanations are provided.

Code metadata

Current code version	*v1.0*
Permanent link to code/repository used of this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-17-00094
Legal Code License	*MIT*
Code versioning system used	*Git*
Software code languages, tools, and services used	*Python (2.7)*
Compilation requirements, operating environments & dependencies	*Numpy, Sklearn and Gurobi packages must be installed.*
If available Link to developer documentation/manual	*None*
Support email for questions	ecimen@anadolu.edu.tr

1. Motivation and significance

Extraction of meaningful information from data is a critical step in many modern applications [1]. Data mining, clustering and classification tools are the key technologies in such problems including gesture, face, finger print, and character recognition. A typical approach is to split the problem into feature extraction and classifier optimization steps, although more integrated mining approaches exist. The focus of this research is towards the "classification" problem. A sub-class of classifiers can be put in a category of mathematical programming-based classifiers, involving linear or integer programming during classifier optimization. In [2] a linear separation algorithm was developed by solving LP. In [3] hyper-planes were used as separating surfaces. In [4] the concept Max–Min separation was introduced by defining non-smooth non-convex error function. Usage of non-smoothness in classification was discussed in [5]. Other usages of optimization in classification literature can be found in [6–9].

Roots of this paper depends on Polyhedral Conic Functions (PCFs) which was firstly constructed in [1,10]. Using these shapes, various extensions and applications were conducted, such as combination of PCFs with Max–Min separation [11], clustering based PCF algorithm [12] and incremental piecewise linear classification algorithm [13]. Clustering based PCF algorithm was applied to some real-life problems like arrhythmia classification [14] and gesture recognition [15].

This paper provides the algorithmic information, the source, and illustrations about how to use the classifier that is presented in detail in the DSP part of this special issue [16]. We name the proposed improvement as "Incremental Conic Functions" (ICF) algorithm. It is developed to improve k-Means based PCF algorithm [12] by avoiding two drawbacks. First drawback of k-Means based PCF algorithm was the necessity of initially determining number of clusters, k. The new ICF algorithm does not require a guess, and finds number of clusters incrementally. Second drawback of k-Means based PCF algorithm was about the size of the problem to

* Corresponding author.
E-mail address: ecimen@anadolu.edu.tr (E. Cimen).

https://doi.org/10.1016/j.softx.2017.12.003
2352-7110/

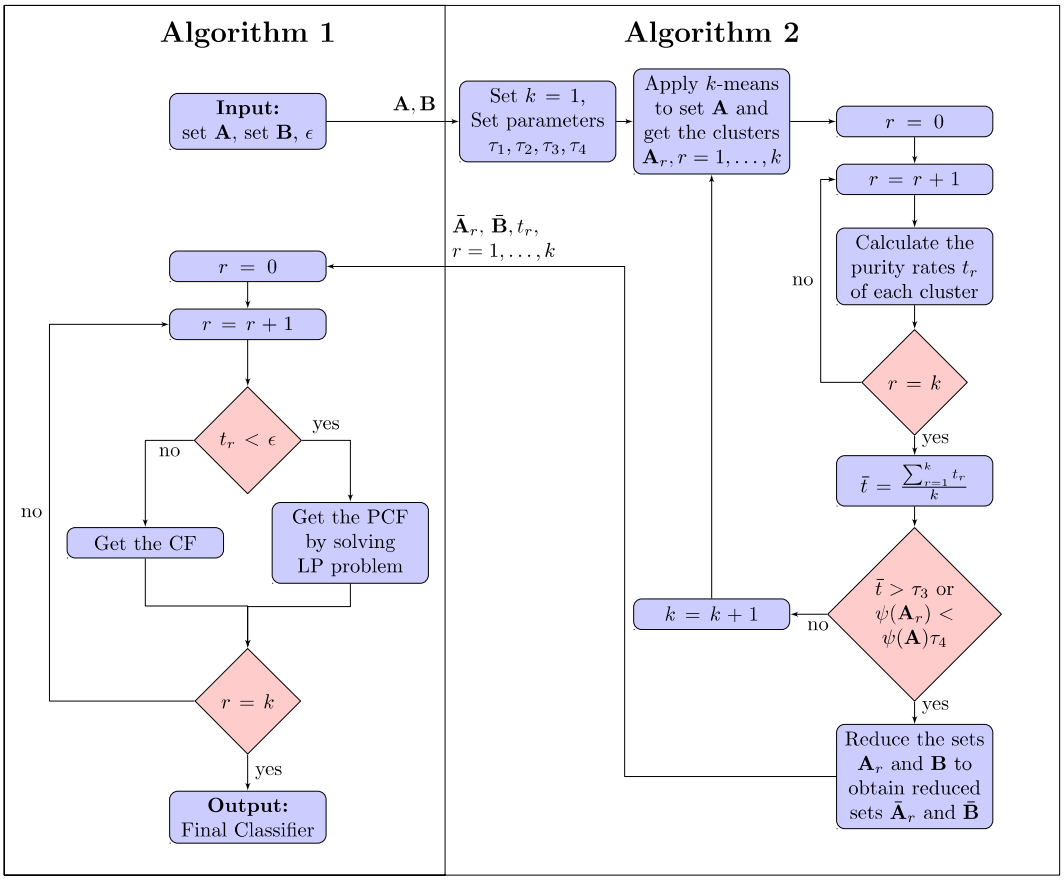

Fig. 1. The flowchart of the ICF Algorithm [16].

be solved with linear programming (LP). Related to its structure, *k*-Means based PCF algorithm solves an LP with linear constraints as much as the number of data points. When the data size is large, the model needs to handle too many constraints, causing a time-consuming LP solution. ICF algorithm may not need to solve LP for each conic function; in some cases, it finds conic separation function algebraically, in other cases, it solves LP with fewer constraints by smartly eliminating unnecessary constraints. The above cases were determined according to a parameter called "purity", that is evaluated for each sub-class of datasets. The algorithm either finds the class "pure" to avoid all LP steps and construct a single cone analytically, or finds a region inside the data cluster to be the "largest pure portion" to totally eliminate those data points for faster LP realization of cones only via shell portions of the set.

The added software is the implementation of the ICF algorithm and allows one to find training/test success on datasets related to classification problems. With this software, we have achieved similar training and test results with the previous version, but at about 64% fewer shorter run-times. The software also finds number of clusters by automatically. We claim that ICF algorithm can be used in real-life problems with large datasets.

In Section 2, ICF Algorithm Software is described with pseudo-codes and illustrations. Section 3 explains the software impact and Section 4 concludes. The scientific background of the software is provided in the accompanying DSP part of this Special Issue [16].

2. ICF algorithm software description

This software is composed from three files. With this software, one can use ICF algorithm to classify data. It can read "csv" and "arff" files. Each row represents data points and each column represents features. Last column must indicate class label (numerical). For example, if there are 5 classes in the dataset, class labels must be {1, 2, 3, 4, 5}. Training and test sets can be given separately. For this type of usage, "ICF-Training-Test" file must be used. One can also get *k*-fold cross validation test results and for this type of usage, "ICF-fold" file must be used. Algorithm 2 of ICF is implemented in "ICF-Purity" file. The functions in "ICF-Purity" file are called both from "ICF-Training-Test" and from "ICF-fold". Therefore, program must be run from "ICF-Training-Test" or "ICF-fold" according to test type. ICF algorithm is implemented with Python 2.7 and shared with the MIT license. Requirements of this software are Numpy, Sklearn and Gurobi Python packages. A detailed user guide can be found from the software link.

The ICF algorithm is a nesting combination of the Algorithm 1 and 2. The ICF algorithm flowchart is given in Fig. 1. Algorithm 1 in [16] which is implemented in files "ICF-Training-Test" and "ICF-fold" is given as Algorithm 1:

Algorithm 2 in [16] which is implemented in file "ICF-Purity" is given as Algorithm 2:

Mathematical details of Algorithm 1 and Algorithm 2 are given in the corresponding DSP article [16].

2.1. Illustrative example

In this subsection, we present an illustrative example to explain the algorithm better via a sample dataset [12]. One can see from Fig. 2 that convex hulls of all classes are overlapping. Moreover, green-labeled class has two separate subsets. This makes classification harder.

Firstly, training and test sets are determined. After that the first class which is going to be classified is selected (set *A*: yellow-labeled in Fig. 2) and the other classes are considered as set *B*. Black

Algorithm 1:

Step	*Action*
0	Read dataset and choose $\epsilon \in [0,1]$
1	Apply ***Algorithm 2*** using set $\mathbf{A}$ and set $\mathbf{B}$ to get: the set $\mathbf{A}_r$, reduced sets as $\overline{\mathbf{A}}_r$, $\overline{\mathbf{B}}$, purity rates of $\mathbf{A}_r$ as t_r and cluster centroids for each cluster as $\mathbf{c}_r$. $r \in \{1,\ldots,k\}$

2	**For each** r **in** {1,...,k}
	If $t_r < \epsilon$ **then**
	Obtain conic function for the set $\mathbf{A}_r$ algebraically as $g_r(\mathbf{x})$ with the parameter γ_r
	Else
	Construct the LP model with the reduced data sets $\overline{\mathbf{A}}_r$, $\overline{\mathbf{B}}$ and $\mathbf{c}_r$ as LP_r.
	Solve LP_r to get corresponding PCF as $g_r(\mathbf{x})$ with parameters $(\mathbf{w}^r, \xi_r, \gamma_r, \mathbf{c}_r)$

3	Obtain final classifier as point wise minimum of all cluster's functions as $g(\mathbf{x}) = \min_{r=1,\ldots,k} g_r(\mathbf{x})$
4	Get Training -Test success and computation time.

Algorithm 2:

Step	*Action*
0	Set $k = 1$ and choose $\tau_1, \tau_2 \in [0,\infty]$, $\tau_3, \tau_4 \in [0,1]$. Define: $\psi\{\cdot\}$ operator as the number of data points in the corresponding set
1	Apply k-Means Algorithm to $\mathbf{A}$. Get clusters, $\mathbf{A}_r$ and cluster centers $\mathbf{c}_r$ for each $r \in \{1,\ldots,k\}$

2	**For each** r **in** {1,...,k}
	calculate the purity rate of $\mathbf{A}_r$ as t_r.
	If mean value of cluster purities $\bar{t} > \tau_3$ or $\psi\{\mathbf{A}_r\} < \psi\{\mathbf{A}\} \times \tau_4$ **then**
	go to Step *3*.
	Else
	increment $k = k + 1$
	go to *Step 1*.

3	Calculate reduced sets $\overline{\mathbf{A}}_r$ and $\overline{\mathbf{B}}$
4	Return reduced sets $\overline{\mathbf{A}}_r$ and $\overline{\mathbf{B}}$ cluster centers $\mathbf{c}_r$ purities t_r

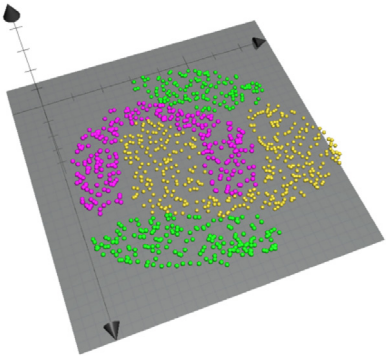

Fig. 2. Illustrative example dataset with 3 classes in R^2 [12]. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

points belong to *B* in Fig. 3. After that, radii of set *A* are calculated and used to find purity of set *A*. If the purity is not greater than a pre-set threshold, set *A* is divided to two clusters with *k*-Means algorithm: labeled as dark- and light-yellow (Fig. 3).

Assume that the purity of the light-yellow labeled cluster in Fig. 3 is not pure enough. Then data elimination of ICF algorithm occurs as eliminating points inside that light-yellow labeled cluster which corresponds to the largest radius with exact purity. In Fig. 4, notice that the central portion of the light-yellow cluster is eliminated. This reduces the conic function classifier programming to run much faster.

Another case is for the green-labeled class of the same dataset. The data in this class is split into three regions, so $k = 3, r \in \{1, 2, 3\}$ for Algorithm 1—Step 2. Fig. 5 shows two pure subsets (a and b), where ideal cones are algebraically evaluated with no LP iterations. However, the bottom part of this green-labeled class is not pure, and after eliminating its pure center, LP is necessary to combine several cones to form a classification as given in Fig. 5(c).

Please cite this article in press as: Cimen E., et al. ICF: An algorithm for large scale classification with conic functions. SoftwareX (2017), https://doi.org/10.1016/j.softx.2017.12.003.

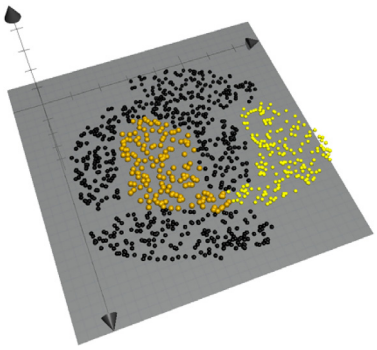

Fig. 3. Two sub-clusters of set A: dark- and light-yellow [16]. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

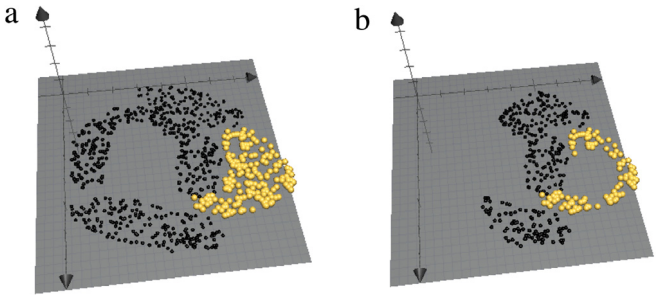

Fig. 4. Data elimination for light-yellow labeled cluster. (a) before elimination, (b) after elimination [16]. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The combination of classifiers for green-labeled class is illustrated in Fig. 6 (by executing Step 3 of Algorithm 1). For this example, eventual classifier $g\,(x)$ for green-labeled class is obtained as point-wise minimum of conic functions found in Algorithm 1—Step 2.

$$g\,(x) = \min\{g_1\,(x)\,, g_2\,(x)\,, g_3(x)\}\,.$$

3. Impact

The main motivation of this paper is to present a successful and fast classifier (PCF) to the signal processing and data analysis communities, whose theoretical foundations are presented in the accompanying DSP special issue [16]. PCF is a new method that

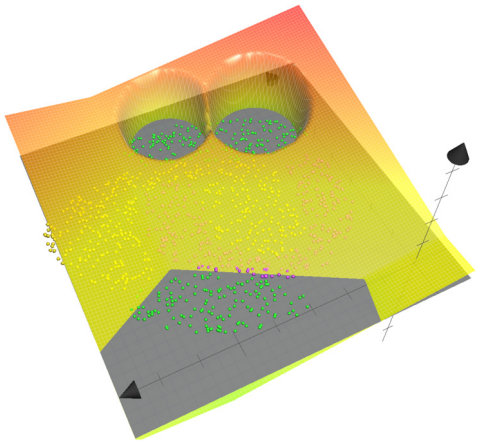

Fig. 6. Combination of two algebraic and one LP-based conic functions according to Step 3 of Algorithm 1 [16]. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

successfully competes against popular classifiers, including the celebrated SVMs. One drawback of PCFs (and its application variant of *k*-means based PCF) was its certain level of computational requirements. With this work, two novel computational improvements were introduced (as explained in Section 1, 2 and [16]). The improvements are observed to keep the classification accuracy whilst reducing the execution time by 64% (according to the experimented datasets). It is argued that, for all possible applications that require classification, the proposed ICF methodology is expected to render a plausible alternative. Now, with its even faster implementation, scientist and practicing engineers are encouraged to put the supplied algorithm to their suit of classifiers among other classical classifiers such as Naïve Bayes, SVM, etc.

In order to show the efficiency and the potential of the ICF algorithm, train and test set accuracy rates are presented in Table 1. The *k*-Means based PCF algorithm is a novel method which outperforms several state-of-the-art classifiers, including SVM [12]. One can see from the Table 1 that the ICF algorithm achieves the same or higher accuracy rates with the *k*-Means based PCF algorithm in much less time.

4. Conclusions

The presented software (ICF) is implemented to improve the already successful PCF classifier for possible applications that require processing of large datasets. The improvement is achieved by

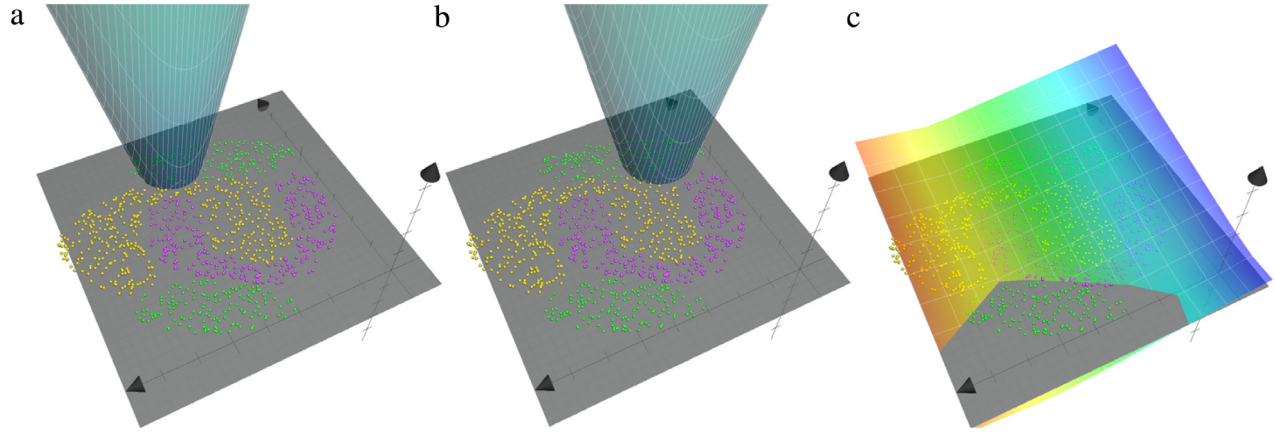

Fig. 5. Classifiers that are obtained from LP. (a) $g_1\,(x)$ (b) $g_2\,(x)$, (c) $g_3\,(x)$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Please cite this article in press as: Cimen E., et al. ICF: An algorithm for large scale classification with conic functions. SoftwareX (2017), https://doi.org/10.1016/j.softx.2017.12.003.

Table 1
Train and test set accuracy rates of the ICF Algorithm on real life dataset, together with run-times [16].

Dataset	Proposed algorithm			*k*-means based PCF			
	Train	Test	Time (s)	*k*	Train	Test	Time (s)
Abalone [17]	48.80	47.70	16.02	5	48.50	46.7	40.90
Page blocks [18]	89.10	86.08	27.30	3	93.75	81.13	98.00
Satellite [19]	87.40	86.10	181.50	6	87.60	86.10	372.20
Shuttle [20]	93.48	93.47	2671.20	3	96.42	96.50	13038.50
Cover type [21]	61.43	53.60	1242.33	4	67.78	52.45	2972.08

methodologically eliminating (i) several data points and (ii) occasionally the necessity of conic function iteration. The reduced data point count causes a simpler (hence faster) optimization problem. In certain cases, where the purity of a subclass is good enough, there remains no need to iteratively combine conic shapes, but a single analytic cone representation suffices for classification of that pure group. This observation also greatly contributes to the speed of the algorithm. The proposed method is observed to be, on average, 3-times faster than the current PCF implementation [16]. A final superiority of the proposed method is that it methodologically obtains the number of sub clusters (k of the k-means step), which should previously be supplied to the algorithm in the previous implementation of k-Means based PCF [13]. Together with these improvements, we introduce a usable software implementation to scientists and practicing engineers that require state of the art classification over their large datasets.

Acknowledgment

This paper is supported by Anadolu University Scientific Research Projects Commission with project numbers 1506F499, 1603F122, 1605F524 and 1605F435.

References

[1] Ozturk G. A new mathematical programming approach to solve classification problems. [Ph.D. thesis], Eskisehir Osmangazi University, Institute of Science; 2007 in Turkish.
[2] Bennett KP, Mangasarian OL. Robust linear programming discrimination of two linearly inseparable sets. Optim Methods Softw 1992;1(1):23–34.
[3] Astorino A, Gaudioso M. Polyhedral separability thorough succesive LP. J Optim Theory Appl 2002;112(2):265–93.
[4] Bagirov AM. Max–min separability. Optim Methods Softw 2005;20(2–3):271–90.
[5] Astorino A, Fuduli A, Gorgone E. Non-smoothness in classification problems. Optim Methods Softw 2008;23(5):675–88.
[6] Astorino A, Fuduli A, Astorino M. Nonlinear programming for classification problems in machine learning. In: AIP conference proceedings-numerical computations: theory and algorithms, vol. 1776, No. 1, Pizzo Calabro, Italy, 18–25 June 2016, 2016. p. 1–4.
[7] Astorino A, Fuduli A. Support vector machine polyhedral separability in semisupervised learning. J Optim Theory Appl 2015;164(3):1039–50.
[8] Astorino A, Gaudioso M. Ellipsoidal separation for classification problems. Optim Methods Softw 2005;20(2–3):267–76.
[9] Huang JAK Suykens, Wang S, Hornegger J, Maier A. Classification with truncated ℓ_1 distance kernel. IEEE Trans Neural Netw Learn Syst 2017;PP(99):1–6.
[10] Gasimov R, Ozturk G. Separation via polyhedral conic functions. Optim Methods Softw 2006;21(4):527–40.
[11] Bagirov A, Ugon J, Webb D, Ozturk G, Kasımbeyli R. A novel piecewise linear classifier based on polyhedral conic and max–min separabilities. TOP 2013;21(1):3–24.
[12] Ozturk G, Ciftci T. Clustering based polyhedral conic functions algorithm in classification. J Ind Manag Optim 2015;11(3):921–32.
[13] Ozturk G, Bagirov AM, Kasimbeyli R. An incremental piecewise linear classifier based on polyhedral conic separation. Mach Learn 2015;101(1/3):397–413.
[14] Cimen E, Ozturk G. Arrhythmia classification via k-means based polyhedral conic functions algorithm. In: 2016 International conference on computational science and computational intelligence, NV, USA, December 14–17, 2016. p. 798–802.
[15] Cimen E. Gesture recognition with polyhedral conic functions based classifiers. [M.Sc. thesis], Anadolu University, Institute of Science; 2013 in Turkish.
[16] Cimen E, Öztürk G, Gerek ÖN. Incremental conic functions algorithm for large scale classification problems. Digit Signal Process (SI on Rep. Research) 2017. http://dx.doi.org/10.1016/j.dsp.2017.11.010. [to appear].
[17] Abalone data set, https://archive.ics.uci.edu/ml/datasets/Abalone.
[18] Page block classification data set, https://archive.ics.uci.edu/ml/datasets/Page+Blocks+Classification.
[19] Statlog (Landsat satellite) data set, https://archive.ics.uci.edu/ml/datasets/Statlog+(Landsat+Satellite).
[20] Statlog (Shuttle) data set, https://archive.ics.uci.edu/ml/data,sets/Statlog+(Shuttle).
[21] Covertype data set, https://archive.ics.uci.edu/ml/datasets/Covertype.