Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2015) 1–8

Procedia Computer Science

www.elsevier.com/locate/procedia

# Active Manifold Learning with Twitter Big Data

Catarina Silva[a,c], Mário Antunes[a,b], Joana Costa[a,c], Bernardete Ribeiro[c]

[a] *School of Technology and Management Polytechnic Institute of Leiria, Portugal*
[b] *Center for Research in Advanced Computing Systems University of Porto,Portugal, mantunes@dcc.fc.up.pt*
[c] *Center for Informatics and Systems University of Coimbra, Portugal, {catarina,bribeiro,joanamc}@dei.uc.pt*

**Abstract**

The data produced by Internet applications have increased substantially. Big data is a flaring field that deals with this deluge of data by using storage techniques, dedicated infrastructures and development frameworks for the parallelization of defined tasks and its consequent reduction. These solutions however fall short in online and highly data demanding scenarios, since users expect swift feedback.

Reduction techniques are efficiently used in big data online applications to improve classifications problems. Reduction in big data usually falls into the following two main methods: (i) to reduce the dimensionality by pruning or reformulating the feature set; and (ii) to reduce the sample size by choosing the most relevant examples. Both approaches have benefits, not only of time consumed to build a model, but eventually also performance-wise, usually by reducing overfitting and improving generalization capabilities.

In this paper we investigate reduction techniques that tackle both dimensionality and size of big data. We propose a framework that combines a manifold learning approach to reduce dimensionality and an active learning SVM-based strategy to reduce the size of labeled sample. Results on Twitter data show the potential of the proposed active manifold learning approach.



*Keywords:* Big data, Support Vector Machine, Manifold, Twitter

## 1. Introduction

Big data is one of the major trends in research in the last years and it is expected that science, business, industry, government, society, etc. will undergo a thorough change with the influence of big data [1]. Although one might argue that we have been in the presence of large data sets for a while and that this new term is just a hype, there are in fact tangible outcomes of this re-branding that are worth analysing, namely by the availability of specific (and free) frameworks [2] like Hadoop (`http://hadoop.apache.org`) or Mahout (`http://mahout.apache.org`).

Big data is a collection of datasets consisting of massive unstructured, semi-structured, and structured data [3]. Big data is being generated by everything around us at all times (http://www.ibm.com/big-data/)d. One of the major sources of data are social networks, e.g. Twitter (`http://twitter.com`), Facebook (`http://facebook.com`) or Instagram (`http://instagram.com`). In this social era, individuals and companies produce enormous amounts of data (*Volume*), extremely heterogeneous (*Variety*) and at alarming rates (*Velocity*). And thus with social networks we get the 3 V's that characterize big data scenarios.

A fourth 'V', for *Veracity*, has been also considered and is in fact extremely important since it relates to the uncertainty, and eventually unavailability, of data and also to the trust one may or may not put on big

data information. Specially when dealing with social networks big data, it may became crucial. Problems related to *Veracity* usually result in few reliably classified examples, which poses barriers to supervised learning approaches, and tends to lead to the need for the use of unlabeled data and active learning. Privacy is also a challenge, since one must always consider to use only publicly available data or data explicitly provided for the goal at hand. Given this setup, data scientists are in high demand and practical results are becoming extremely valuable research and business-wise. An example can be found in [4] where a distributed strategy with decision trees and Support Vector Machines (SVM) is proposed to predict the price trends of stock futures with large amounts of data. The focus was on the proposal of statistical features which were achieved using MapReduce algorithm [5].

Putting more emphasis on representation, [3] proposes a unified tensor model to represent the unstructured, semi-structured, and structured data where various types of data are represented as subtensors and then are merged to a unified tensor. An approach based on singular value decomposition method is introduced to extract information, with competitive results in time complexity, memory usage, and accuracy.

Regarding dimensionality reduction, one can find in [6] an alternative to the usually greedy strategies, by using the Orthogonal Centroid (OC) as feature extraction method that is found very effective in classification problems. Another approach is presented in [7] where a two-step process is proposed to detect forged signatures, first by extracting features from biometric images using a GPU-based SVM classifier. An emerging nonlinear dimension reduction technique is manifold learning [8, 9], which is the process of estimating a low-dimensional structure which underlies a collection of high-dimensional data, bringing several advantages, namely visualization capabilities, as we will show in this work.

Nevertheless these cutting edge applications, challenges arise in online scenarios when using such robust frameworks. When searching information from an online source like Twitter, reducing size and dimension in supervised learning has gained interest in the machine learning community as a way to reduce time spent constructing learning models, but also as an effective way of improving performance by pruning extraneous data. In fact, dimensionality reduction has been considered as an essential data preprocessing technique for large-scale and streaming data classification tasks [6]. This appeal is underpinned by the tremendous increase of digital information that often leads applications and learning algorithms to include a dimension/size reduction step. High dimensionality has usually at least two perspectives. On one hand, the number of examples is massive and the difficulty to keep a representative training set of labeled instances is growing. On the other hand, the representation of each example can also reach high dimensions and make the decision space more complex in applications like text classification or gene expression.

In this paper we propose a framework to reduce size and dimension in Twitter big data. Size is reduced by using a SVM active learning strategy that takes place after a manifold reduction step is put forward to reduce the initial huge dimensionality of a text classification problem.

Next two sections will introduce both reductions we are dealing with: dimensionality reduction on Section 2 and size reduction on Section 3. Then, in Section 4 we describe the manifold active learning approach and in Section 5 we show the results obtained along with the experimental setup. Finally, Section 6 discusses conclusions and future work.

## 2. Dimensionality reduction - Manifold Learning

Initial dimensionality reduction is carried out in the feature space as a pre-processing step. Several supervised and unsupervised techniques can be applied. Manifold learning strategies, like Isomap (Isometric Mapping) [10], are effective for extracting nonlinear structures from high-dimensional data in pattern recognition [11]. Finding the structure behind the data may be important for a number of reasons in many applications, such as data visualization. Graphical depiction of the document set can potentially be crucial, since it makes possible to quickly give large amounts of information to a human operator [12]. To this purpose it is appropriately assumed that the data lies on a statistical manifold, or a manifold of probabilistic generative models [13]. It can be regarded as a supervised learning method, where the training labels play a central role. In such a scenario, manifold learning can be used not only with the traditionally associated algorithms, such as K-Nearest Neighbors (K-NN), but also with state-of-the-art kernel-based machines like SVM [14].

Feature reduction methods aim at choosing from the available set of features a smaller set that more efficiently represents the data. Such reduction is not needed for all classification algorithms as some classifiers are capable of feature selection themselves.

Many approaches have been proposed for dimensionality reduction, such as the well-known methods of Principal Component Analysis (PCA) [15], Independent Component Analysis (ICA) [16] and MultiDimensional Scaling (MDS) [17]. All these methods are well understood and efficient, having thus been widely used in visualization and classification tasks. Unfortunately, they share a common inherent limitation: they are all linear methods while the distributions of most real-world data are nonlinear. In [18] a survey on feature extraction foundations and applications can be found.

Another increasingly used technique is manifold learning [8, 9], which is the process of estimating a low-dimensional structure that underlies a collection of high-dimensional data. Manifold learning can be viewed as implicitly inverting a generative model for a given set of observations [19]. Let $Y$ be a $d$ dimensional domain contained in a Euclidean space $\mathbb{R}^d$. Let $f : Y \rightarrow \mathbb{R}^D$ be a smooth embedding for some $D > d$. The goal of manifold learning is to recover $Y$ and $f$ given $N$ points in $\mathbb{R}^D$. Isomap [10] provides an implicit description of the mapping $f$ (or $f^{-1}$). Given $X = \{\mathbf{x}_i \in \mathbb{R}^D | i = 1 \dots N\}$ find $Y = \{\mathbf{y}_i \in \mathbb{R}^d | i = 1 \dots N\}$ such that $\{\mathbf{x}_i = f(\mathbf{y}_i) | i = 1 \dots N\}$. Without imposing any restrictions of $f$, the problem is ill-posed. The simplest case is a linear isometry, i.e. $f$ is a linear mapping from $\mathbb{R}^d \rightarrow \mathbb{R}^D$, where $D > d$.

In Isomap [10] the local neighborhood of each example is preserved, while trying to obtain highly nonlinear embeddings with manifold learning. For data lying on a nonlinear manifold, the *true distance* between two data points is the geodesic distance on the manifold, i.e. the distance along the surface of the manifold, rather than the straight-line Euclidean distance. The main purpose of Isomap is to find the intrinsic geometry of the data, as captured in the geodesic manifold distances between all pairs of data points. The approximation of geodesic distance is divided into two cases. In case of neighboring points, Euclidean distance in the input space provides a good approximation to geodesic distance. In case of faraway points, geodesic distance can be approximated by adding up a sequence of *short hops* between neighboring points. Isomap shares some advantages with PCA and MDS, such as computational efficiency and asymptotic convergence guarantees, but with more flexibility to learn a broad class of nonlinear manifolds. The Isomap algorithm takes as input the distances $d(\mathbf{x}_i, \mathbf{x}_j)$ between all pairs $\mathbf{x}_i$ and $\mathbf{x}_j$ from $N$ data points in the high-dimensional input space. The algorithm outputs coordinate vectors $\mathbf{y}_i$ in a $d$-dimensional Euclidean space that best represent the intrinsic geometry of the data. Isomap is accomplished following these steps:

1. Construct the neighborhood graph: Define the graph $G$ over all data points by connecting points $\mathbf{x}_i$ and $\mathbf{x}_j$ if they are closer than a certain distance $\varepsilon$, or if $\mathbf{x}_i$ is one of the $K$ nearest neighbors of $\mathbf{x}_j$. Set edge lengths equal to $d(\mathbf{x}_i, \mathbf{x}_j)$.
2. Compute shortest paths: Initialize $d_G(\mathbf{x}_i, \mathbf{x}_j) = d(\mathbf{x}_i, \mathbf{x}_j)$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are linked by an edge; $d_G(\mathbf{x}_i, \mathbf{x}_j) = +\infty$ otherwise. Then for each value of $k = 1, 2, \dots, N$ in turn, replace all entries $d_G(\mathbf{x}_i, \mathbf{x}_j)$ by $min\{d_G(\mathbf{x}_i, \mathbf{x}_j), d_G(\mathbf{x}_i, \mathbf{x}_k) + d_G(\mathbf{x}_k, \mathbf{x}_j)\}$. The matrix of final values $\mathbf{D}_G = \{d_G(\mathbf{x}_i, \mathbf{x}_j)\}$ will contain the shortest path distances between all pairs of points in $G$.
3. Apply MDS to the resulting geodesic distance matrix to find a $d$-dimensional embedding.

This is an unsupervised procedure and constitutes a preprocessing step for classification. Basically it performs a transformation from a high dimensional input data space into a lower dimensional feature space. Then a classifier can be applied to the resulting data. However, the mapping function given by Isomap is only implicitly defined. Therefore, it should be learned by nonlinear interpolation techniques, such as generalized regression neural networks [20], which can then transform the new test data into the reduced feature space before prediction.

In the supervised version of Isomap [21], the information provided by the training class labels is used to guide the procedure of dimensionality reduction. The training labels are used to refine the distances between inputs. The Euclidean distance $d(\mathbf{x}_i, \mathbf{x}_j)$ between two given observations $\mathbf{x}_i$ and $\mathbf{x}_j$, labeled $y_i$ and $y_j$ respectively, is replaced by a dissimilarity measure [21]:

$$D(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} \sqrt{1 - e^{\frac{-d^2(\mathbf{x}_i, \mathbf{x}_j)}{\beta}}} & y_i = y_j, \\ \sqrt{e^{\frac{-d^2(\mathbf{x}_i, \mathbf{x}_j)}{\beta}}} - \alpha & y_i \neq y_j. \end{cases} \tag{1}$$

Note that the Euclidean distance $d(\mathbf{x}_i, \mathbf{x}_j)$ is in the exponent and the parameter $\beta$ is used to avoid that $D(\mathbf{x}_i, \mathbf{x}_j)$ increases too rapidly when $d(\mathbf{x}_i, \mathbf{x}_j)$ is relatively large. Hence, $\beta$ depends on the *density* of the data set and is usually set to the average Euclidean distance between all pairs of data points. On the other hand, $\alpha$ gives a certain possibility to points in different classes to be *closer*, i.e. to be more similar, than those in the same class. $\alpha$ must be chosen so that the dissimilarity is never negative, at most it can be zero. This procedure allows a better determination of the relevant features and will definitely improve visualization [22].

## 3. Size reduction - Active Learning

To reduce the number of labeled training examples needed for a supervised learning algorithm, such as support vector machines (SVMs), there have been many studies employing unlabeled documents in the learning task, like, transductive learning [23], co-training [24] and active learning [25, 26, 27, 28, 29]. Usually, the training set is chosen to be a random sampling of instances. However, in many cases active learning can be employed. Here, the learner can actively choose the training data. It is hoped that allowing the learner this extra flexibility will reduce the learner's need for large quantities of labeled data and hence reduce training time [28, 30]. Pool-based active learning for classification was introduced by Lewis and Gale [25]. The learner has access to a pool of unlabeled data and can request the true class label for a certain number of instances in the pool.

To achieve the best classification performance on big data with a machine learning technique, one can face two problems: not enough data or too much data. Active learning mechanisms can be applied in both scenarios:

1. When there is not enough labeled data or there is labeled data but it is not reliable (*Veracity* in big data), and a large set of unlabeled data is readily available;
2. When there is too much labeled data (*Volume* and *Velocity* in big data) and algorithms can benefit if a selection is carried out.

Any active learning algorithm selects from a pool of examples which should be used (usually after being classified) to create the learning model. Hence, to actively learn we aim at selecting those examples that, when labeled and incorporated into training, will minimize classification error over the distribution of future examples. The main issue with active learning is to find a way to choose good requests or queries from the pool. It is assumed that the instances $\mathbf{x}$ are independent and identically distributed (i.i.d.) according to some underlying distribution $F(\mathbf{x})$ and the labels are distributed with some conditional distribution [31].

In this work we propose an SVM-based active learning strategy. In SVMs, Support Vectors (SVs) and weights define the model. SVs define the optimal separating hyperplane (OSH) [14]. It is well-known that the examples in the margin are more informative because the uncertainty associated is higher, thus the most informative unlabeled examples are potentially those closer to any of the existing SV in the model, since they can potentially alter the OSH. To define these examples we propose a kernel-based approach, that defines a design matrix $\mathbf{\Psi}$, assessing the distances between the existing SV and the set of unlabeled examples available.

Given an initial SVM model, induced using input-output labeled training data $(\mathbf{x}_1, y_1), ...(\mathbf{x}_l, y_l) \in \mathbb{R}^M \times \{\pm 1\}$, resulting in $(\boldsymbol{\rho}_1, ..., \boldsymbol{\rho}_s) \in \mathbb{R}^M$ support vectors. Given also unlabeled data $\mathbf{U}$, $(\mathbf{u}_1, ..., \mathbf{u}_h) \in \mathbb{R}^M$, the similarity between an SV and an unlabeled document is defined as

$$\Psi_{ij} = k(\boldsymbol{\rho}_i, \mathbf{u}_j), \tag{2}$$

where $k$ represents the kernel used to define a higher dimension space where points can be compared. For a generic kernel function $\Phi$, $\Psi_{ij}$ is the dot product

$$\Psi_{ij} = \langle \Phi(\boldsymbol{\rho}_i), \Phi(\mathbf{u}_j) \rangle. \tag{3}$$

Assuming a linear kernel, $\Psi_{ij}$ is

$$\Psi_{ij} = \langle \boldsymbol{\rho}_i, \mathbf{u}_j \rangle. \tag{4}$$

For a linear kernel the design matrix is simplified

$$\mathbf{\Psi}_{linear} = \boldsymbol{\rho} \,.\, \mathbf{U}'. \tag{5}$$

After this design matrix is constructed, it remains to be determined which unlabeled examples are potentially more informative. The procedure is easily implemented as follows. First, the closest SV to any test unlabeled example is given by the max value of each column in the design matrix (6). Second, a definable number of unlabeled examples with smaller minimum distance to an SV are chosen and added to the training set.

$$\left[\max(k(\boldsymbol{\rho}_i, \mathbf{u}_1)) \quad \ldots \quad \max(k(\boldsymbol{\rho}_i, \mathbf{u}_h))\right]. \tag{6}$$

## 4. Proposed approach

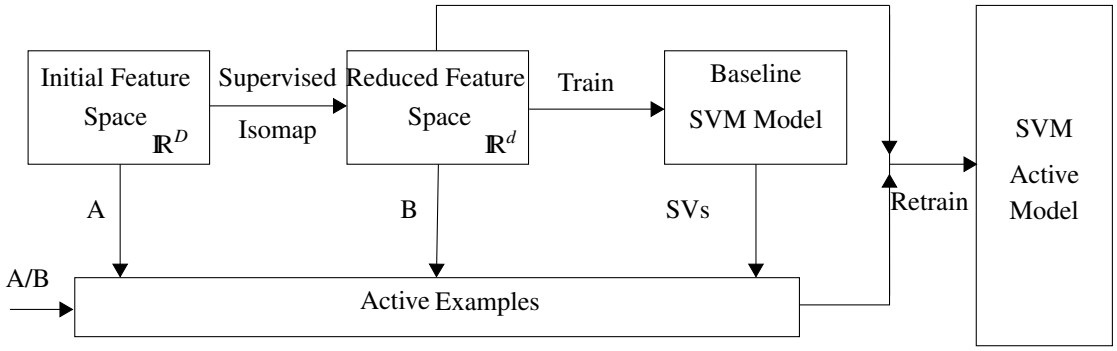



Fig. 1. Active learning strategy.

Fig. 1 depicts the proposed active learning approach for learning with Twitter Big Data. Analyzing the figure one can identify three stages before constructing the SVM Active Model: initial feature space, reduced feature space and baseline SVM model. Additionally, one can also distinguish a switch that defines two functioning modes: A and B that correspond to different active strategies: Active A and Active B.

The first step between initial and reduced feature space is carried out through a manifold reduction stategy based on supervised Isomap [21]. In this step the features are constructed using the training labels, hence the supervised nature of the process. A dissimilary measure (1) is used to improve the baseline Isomap distance using label information, with $\alpha$ taking the value of 0.65 and $\beta$ the average Euclidean distance between all pairs of text data points.

When a reduced space is reached, our aim is to learn a kernel-based model that can be applied to unseen examples. We propose an active learning support vector machine (SVM) with a linear kernel, commonly used and adequate in text classification problems.

Although this model is straightforward for training, Isomap does not provide an explicit mapping of examples. Therefore we can not generate the test set directly, since we would need to use the labels. To circumvent this hurdle, we use a generalized regression neural network (GRNN) [20] to learn the mapping and apply it to each test document, before the SVM prediction phase.

To determine the active examples presented in the bottom of Fig. 1, i.e. the unlabeled examples that are potentially more informative the design matrix (see Section 3) can be constructed in the original $\mathbb{R}^D$ space or in the reduced $\mathbb{R}^d$ space.

The two functioning modes (A and B) refer to whether active examples are chosen from initial feature space (Active A) or from the reduced feature space (Active B). However, mode A (adding examples from $\mathbb{R}^D$) is computationally more intensive, while strategy B is simpler. To choose active learning examples from the original feature space, first the baseline SV have to be remapped back into $\mathbb{R}^D$, then the design matrix is constructed and the active examples chosen. Before the final learning procedure can take place, a new Isomap feature reduction step with these new examples is carried out.

On the other hand, mode B that chooses the examples directly from the reduced feature space includes a more complex initial Isomap step, with potential active examples, but does not include other overheads.

## 5. Experimental Setup

To test the proposed framework we will use a real big data dataset retrieved from the Twitter public stream. In this section we describe the dataset used, the pre-processing and classification methods applied to Twitter messages retrieved. We conclude by describing the experimental results obtained and delineating the main conclusions observed.

### *5.1. Problem Description: Twitter classification*

Twitter messages are termed *tweets*. Each tweet is up to 140 characters long and is usually labeled with a term preceded with the symbol "#", called *hashtag*. The following is a rough example of a tweet labeled with the hashtag `#nowplaying`: `#nowplaying tow waits face to the highway`. The datasets used to evaluate the proposed active manifold learning strategy were built from the public Twitter stream, which is available through the public API (https://dev.Twitter.com). The tweets were collected between 28th December 2014 and 21th January 2015, only including English written tweets. The API was retrieved for the following hashtags: `#bieber`, `#jobs` and `#syrisa`. These hashtags were chosen to represent different types of tweets that represent different trends. To handle such a multi-class problem we adopted a one-against-all approach.

Before learning, the hashtag was removed from the message content and became exclusively used as the document label (class) for classification purposes. The resulting dataset only includes valid tweets, i.e. those which contain a message content besides the hashtag. Each one of the 3 considered classes (represented by the hashtags) were organised in two datasets using chronological order in terms of the date the tweet had been posted:

1. Big split: 1800 tweets, each class has 600 tweets, 70% were used for training (420 of each class) and 30% for testing (180 of each class);
2. Small split: the training set is defined for each category by randomly selecting 10 positive and 10 negative examples and the testing set is exactly the same for the sake of comparison [32].

A tweet is represented as a document in which its words are the collection of features, built as the dictionary of unique terms presented in the documents collection. Each tweet is a vector with one element for each term occurring in the whole collection. The weighting scheme used to represent each term is the *term frequency - inverse document frequency*, also known as *tf-idf*.

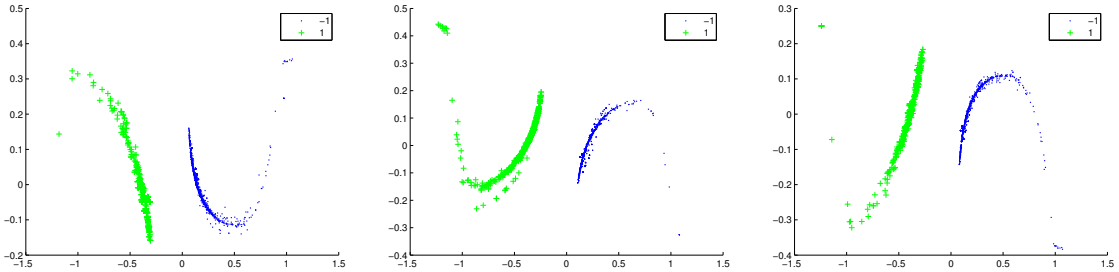
Pre-processing methods were applied in order to reduce feature space given the usual dimensionality associated with text classification and big data, to reduce the size of the document representation and to prevent mislead classification of some words. Some examples of such methods are the removal of *stop-words*, such as articles, prepositions and conjunctions, as well as some non-informative words that appear more frequently than other informative ones. Stemming was also applied by removing case and inflection information of each word, reducing it to the word root.

### *5.2. Evaluation metrics*

The performance will be evaluated using the testing sets defined for each category, with several metrics to determine the learning ability. In order to assess a binary decision task, we first define a contingency matrix representing the possible outcomes of the classification. Several measures were defined based on the results of the contingency table, such as, Error Rate ($\frac{b+c}{a+b+c+d}$), Recall ($\frac{a}{a+c}$) and Precision ($\frac{a}{a+b}$), where $a$, $b$, $c$ and $d$ represent the true positives, false positives, false negatives and true negatives respectively. Measures that combine recall and precision have been defined, such as, the van Rijsbergen's $F_\beta$ measure [33], which combines recall and precision in a single score, $F_\beta = \frac{(\beta^2+1)P\times R}{\beta^2 P+R}$. This measure is one of the best suited measures for text classification [34], thus results reported in this paper are macro-averaged F1 values.

### *5.3. Experimental Results and Discussion*

Fig. 2 shows the results of manifold learning applied to the Twitter dataset described in Section 5.1. As can be gleaned from the figures, a major benefit arising from the use of manifold learning to reduce the feature space dimension is the possibility of showing the clear separation of classes. This potential of



(a) #bieber (b) #jobs (c) #syrisa

Fig. 2. Separation of classes in the reduced dimension space.

visualization can be extremely important in real applications, since big data is being increasingly applied in numerous fields of research and business and users become more comfortable with classifications that support their decisions when they can visualize the result.

Tables 1 and 2 present the F1 performance results for Big and Small splits respectively.

| | Baseline | Active A | Active B |
|---|---|---|---|
| #bieber | 95.18% | 95.18% | 97.73% |
| #jobs | 92.54% | 92.54% | 92.90% |
| #syrisa | 95.40% | 95.40% | 95.07% |
| **Average** | 94.37% | 94.37% | 95.22% |

Table 1. Performances with Small split for both approaches.

| | Baseline | Active A | Active B |
|---|---|---|---|
| #bieber | 73.03% | 84.25% | 80.36% |
| #jobs | 73.68% | 88.96% | 94.49% |
| #syrisa | 73.68% | 88.96% | 94.49% |
| **Average** | 73.81% | 90.31% | 86.75% |

Table 2. Performances with Big split for both approaches.

An initial analysis of both tables let one know that both proposed active approaches generally improve the baseline SVM classification results for both defined splits. This improvement is more evident for the Small split, since its initial performance is understandably weaker, given the reduced number of training examples used. In this Small split scenario the use of active learning can make a serious difference and results show that Active A approach is significantly better.

Regarding the Big split the improvement is slight and only visible in the Active B approach. Nevertheless, given the strong baseline results, product of the learning in the reduced manifold feature space, even such improvement can prove to be relevant in specific classification tasks.

## 6. Conclusions and Future Work

In this paper we proposed a framework to cope simultaneously with the problem of reducing the size and dimension in big data supervised learning settings.

Feature space reduction is achieved by generating a statistical manifold to suit the data through a supervised version of Isometric Mapping. This reduction makes it possible to visualize the decision space using the manifold reduced feature space, giving end users a real sense of confidence in the results. Sample set reduction is obtained by an active learning strategy, based on a kernel trick. Active examples are selected from a large range of examples that are available in big data scenarios. The proposed approach is thus able to deal with high dimensionality in data sets, by both reducing the features and the number of examples needed to reach a desired performance. The proposed framework was applied to a real dataset retrieved from the Twitter public stream which included three hashtags to be predicted. Results were promising and show the robustness of the framework by obtaining good performance for big text datasets processing. Future work will include the expansion of tests to examine in detail the computational improvement achieved by the proposed reduction techniques in larger datasets.

## References


[1] Zhi-Hua Zhou, N. V. Chawla, Yaochu Jin, G. J. Williams, "Big Data Opportunities and Challenges: Discussions from Data Analytics Perspectives", IEEE Computational Intelligence Magazine 9(4), pp. 62–74, 2014.

[2] A. Antoniades, C. C. Took, "A Google approach for computational intelligence in big data", IEEE International Joint Conference on Neural Networks (IJCNN), pp. 1050–1054, 2014.

[3] Liwei Kuang, Fei Hao, L. T. Yang, Man Lin, Changqing Luo, Geyong Min, "A Tensor-Based Approach for Big Data Representation and Dimensionality Reduction", IEEE Transactions on Emerging Topics in Computing 2(3), pp. 280 – 291, 2014.

[4] Dingxian Wang, Xiao Liu, Mengdi Wang, "A DT-SVM Strategy for Stock Futures Prediction with Big Data", 2013 IEEE 16th International Conference on Computational Science and Engineering (CSE), pp. 1005 – 1012, 2013.

[5] Jeffrey Dean, Sanjay Ghemawat, "MapReduce: simplified data processing on large clusters", Communications of the ACM, 51(1), pp. 107–113, 2008.

[6] A. C. Wilkerson, H. Chintakunta, H. Krim, "Computing persistent features in big data: A distributed dimension reduction approach", 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 11–15, 2014.

[7] B. Ribeiro, N.Lopes, J. Goncalves, "Signature identification via efficient feature selection and GPU-based SVM classifier", IEEE International Joint Conference on Neural Networks (IJCNN), pp. 1138–1145, 2014.

[8] Feiping Nie, Dong Xu, Tsang I.W.-H., Changshui Zhang, "Flexible Manifold Embedding: A Framework for Semi-Supervised and Unsupervised Dimension Reduction", IEEE Transactions on Image Processing, 19(7), pp. 1921–1932, 2010.

[9] Zhang, Zhenyue, Wang, Jing, Zha, Hongyuan, Zhejiang, "Adaptive Manifold Learning", IEEE Transactions on Pattern Analysis and Machine Intelligence, 34(2), pp. 253 – 265, 2012.

[10] J. B. Tenenbaum, V. de Silva and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction", Science, 290(5500), pp. 2319–2323, 2000.

[11] H. Kim, H. Park and H. Zha, "Distance Preserving Dimension Reduction for Manifold Learning", Society for Industrial and Applied Mathematics Int. Conf. Data Mining, vol II, pp. 1147–1151, 2007.

[12] Marcus Butavicius and Michael Lee, "An empirical evaluation of four data visualization techniques for displaying short news text similarities", International Journal of Human-Computer Studies, 65(11), pp. 931–944, 2007.

[13] D. Zhang, X. Chen, and W. Lee, "Text Classification with Kernels on the Multinomial Manifold", ACM Special Interest Group on Information Retrieval 2005, pp. 266–273, 2005.

[14] V. Vapnik, "The Nature of Statistical Learning Theory", 2nd ed, Springer, 1999.

[15] I. T. Jolliffe, "Principal Component Analysis", New York: Springer, 1986.

[16] P. Comon, "Independent Component Analysis: a New Concept?", Signal Processing, 36(3), pp. 287–314, 1994.

[17] Michael Cox and Trevor Cox, "Multidimensional Scaling", Handbook of data visualization, pp. 315–347, Springer, 2008.

[18] Isabelle Guyon and Steve Gunn and Masoud Nikravesh and Lofti Zadeh, "Series Studies in Fuzziness and Soft Computing", Physica-Verlag, Springer, 2006.

[19] Dalton Lunga and Saurabh Prasad and Melba Crawford and Okan Ersoy, "Manifold-Learning-Based Feature Extraction for Classification of Hyperspectral Data: A review of advances in manifold learning", IEEE Signal Processing Magazine, 31(1), pp. 55–66, 2014.

[20] Donald Specht, "A General Regression Neural Network". IEEE Transactions on Neural Networks, 2(6), pp. 568–576, 1991.

[21] X. Geng, De.n Zhan, and Z. Zhou, "Supervised Nonlinear Dimensionality Reduction for Visualization and Classification", IEEE Transactions Systems, Man, and Cybernetics - Part B, 35(6), pp. 1098–1107, 2005.

[22] Silva, C. and Ribeiro, B. ,"Text Classification on Embedded Manifolds", in IBERAMIA, pp. 272-281, 2008.

[23] T. Joachims, "Transductive Inference for Text Classification using Support Vector Machines", *Proceedings of the 16th International Conference on Machine Learning*, ICML, pp. 200–209, 1999.

[24] Yihao Zhang Junhao Wen and Xibin Wang and Zhuo Jiang, "Semi-supervised learning combining co-training with active learning", Journal Expert Systems with Applications, 41(5), pp. 2372–2378, 2014.

[25] D. Lewis, W. Gale, "A sequential algorithm for training text classifiers", *ACM Special Interest Group on Information Retrieval*, pp. 3–12, 1994.

[26] G. Schohn, D. Cohn, "Less is more: Active Learning with Support Vector Machines", *Proceedings of the 17th International Conference on Machine Learning*, ICML, pp. 839–846, 2000.

[27] R.Figueroa and Q. Z.-Treitlerand and L. Ngo and S. Goryachev and E. Wiechmann, "Active learning for clinical text classification: is it better than random sampling?", Journal of the American Medical Informatics Association, 19(5), pp. 809–816, 2012.

[28] S. Tong, D. Koller, "Support vector machine active learning with applications to text classification", *Journal Machine Learning Research*, vol. 2, pp. 45–66, 2001.

[29] Silva, C. and Ribeiro, B. ,"Selecting Examples in Manifold Reduced Feature Space for Active Learning", in IEEE ICMLA pp.54-59, 2008.

[30] E. Pasolli and F. Melgani and D. Tuia and F. Pacifici and W. Emery, "SVM active learning approach for image classification using spatial information", IEEE Transactions on Geoscience and Remote Sensing, 52(4), pp. 2217–2233, 2014.

[31] T. Liu, "Learning to rank for information retrieval", Foundations and Trends in Information Retrieval, 3(3), pp. 225–331, 2009.

[32] C. Silva and B. Ribeiro, "On Text-based Mining with Active Learning and Background Knowledge using SVM", Journal of Soft Computing, Springer, 11(6), pp. 519–530, 2007.

[33] C. van Rijsbergen, "Information Retrieval", Butterworths Ed., 1979.

[34] Miguel Ruiz, Padmini Srinivasan, "Automatic Text Categorization and Its Application to Text Retrieval", *IEEE Tran. Know. Data Eng.*, 11(6), pp. 865–879, 1999.

[35] S. Kaski and J. Nikkila and M. Oja and J. Vennaand P. Törönen and E. Castren, "Trustworthiness and Metrics in Visualizing Similarity of Gene Expression", BMC Bioinformatics, 4(48), 2003.

[36] J. Venna and S. Kaski, "Local Multidimensional Scaling with Controlled Tradeoff between Trustworthiness and Continuity", Workshop of Self-Organizing Maps, pp. 695–702, 2005.