OnlineCM: Real-time Consensus Classification with Missing Values

Bowen Dong* Sihong Xie* Jing Gao† Wei Fan‡ Philip S. Yu*§

Abstract

Combining predictions from multiple sources or models has been shown to be a useful technique in data mining. For example, in network anomaly detection, multiple detectors' output have to be combined to obtain the diagnostic decisions. Unfortunately, as data are generated at an increasingly high speed, existing prediction aggregation methods are facing new challenges. First, the high velocity and hugh volume of the data render existing batch mode prediction aggregation algorithms infeasible. Second, due to the heterogeneity, predictions from multiple models or data sources might not be perfectly synchronized, leading to abundant missing values in the prediction stream. We propose OnlineCM, short for Online Consensus Maximization, to address the above challenges. OnlineCM keeps only a minimal yet sufficient footprint for both consensus prediction and missing value imputation over the prediction stream. In particular, we show that the correlations among base models or data sources are sufficient for effective consensus prediction, require small storage and can be updated in an online fashion. Further, we identify a reinforcing relationship between missing value imputation and the consensus predictions, leading to a novel consensus-based missing values imputation method, which in turn makes model correlation estimation more accurate. Experiments demonstrates that OnlineCM achieves aggregated predictions that has close performance to the batch mode consensus maximization algorithm, and outperforms baseline methods significantly in 4 large real world datasets.

1 Introduction

Ensemble methods for combining the knowledge of multiple data sources and models is fundamental in the era of big data. An important research subject in ensemble learning is on how to aggregate the output of multiple models to achieve better performance. First, it is not uncommon to have multiple data sources available for data mining tasks. The richness of data sources can greatly improve the predictive performance in the tasks. For example, when modeling cellphone users' behaviors for advertising or recommendation, one can utilize and aggregate data sources such as text messages and images, communication networks, location information obtained from GPS, etc. Multiple models can be trained on different data sources and provide complementary perspectives of the data, leading to more robust and accurate predictions. Second, crowdsourcing has emerged as an important information collecting paradigm, where human workers are hired to provide labels of a large amount of data in an much affordable way. Since human annotations can be subjective and noisy, principled ways of annotation aggregation have to be carefully designed to fully utilize the wisdom of the crowd. There has been a great number of existing work on the topics [14, 13, 19, 20, 1, 22, 29, 30]. For example, in [14], they proposed consensus maximization to aggregate the predictions from supervised and unsupervised models using a bipartite graph. In [20], they proposed to use non-negative matrix factorization to aggregate the output of multiple clustering models. In [29], the authors proposed a Bayesian hierarchical model for prediction aggregation.

However, these existing methods fail to address two challenges. First, there are situations where the predictions need to be combined in real time. For example, in online advertisement, a user's profiles, such as age, need to be predicted in real time to deliver certain relevant ads. The predictions of age can come from multiple models or data sources, and we wish to combine these predictions in real time. Most of the existing methods work in the batch mode, and they need to access predictions of all instances from all base models. The only exception is the approach proposed in [13], where the authors extended the consensus maximization algorithm [14] to the online learning setting. However, their update rules fail to fully capture and utilize the model correlations for more accurate prediction aggregation. Second, with more data sources, it is not uncommon to have data that only present in some sources but are missing in other

*University of Illinois at Chicago, Chicago, IL, USA, {bdong5, sxie6, psyu}@uic.edu

†Department of Computer Science, University at Buffalo, Buffalo, NY, USA, jing@buffalo.edu

‡Baidu Big Data Lab, Sunnyvale, CA, USA, wei.fan@gmail.com

§Institute for Data Science, Tsinghua University, Beijing, China

sources. Predictions from individual data sources are therefore tend to have missing values too. Also, since the velocity of the data stream can be so high that some of the base models have to skip some data example to keep up with velocity of the data, and this also introduces missing values. Nonetheless, due to the time constraint, one can not wait for the data or model to be ready to provide the current missing predictions, but have to aggregate the current predictions that are available without synchronization. In sum, base model predictions have to be aggregated with missing values in an online fashion.

In this paper, we propose an algorithm to solve the above two challenges effectively and efficiently. Regarding online prediction aggregation, we analyze the batch mode consensus maximization [14] algorithm and show that the information needed for aggregation is the covariance matrix of the classes/clusters from different base models. Relying on this covariance matrix has two advantages. First, this matrix can be updated online, and updating the matrix when seeing base model predictions for a new instance amounts to cheap bookkeeping operations. Second, we prove that up to any point, this covariance matrix can be updated such that no information is lost comparing to the batch mode consensus maximization that sees the same set of base model predictions. This proves the optimality of the proposed online algorithm in terms of what a batch mode algorithm can achieve.

Although there have been many works on handling missing value, we need to exploit the special structure of the problem studied here. We propose a missing value imputation model that exploits the dependency among base model predictions. In particular, missing predictions from one model are treated as responses while the non-missing predictions from the remaining models serve as variates. A regression model can be built from available complete base model predictions to capture the dependency, and missing values are imputed using the predictions of the regression model. More importantly, the regression model can be chosen to be an online algorithm that can be updated incrementally while maintains an accurate estimation of the prediction dependency. The contribution of the paper are as follows:

- We identify the challenges of online consensus aggregation of the predictions from multiple models or data sources, which finds its applications in online advertisement and recommendation, among others.

- We propose an online aggregation model to optimally capture the information that the batch mode version can upto any time. Missing values are handled by another online learning component that's able to exploit the dependency among base models.

- Experimental results on several big datasets validate the proposed algorithm is . memory and time efficient, and effective in accuracy.

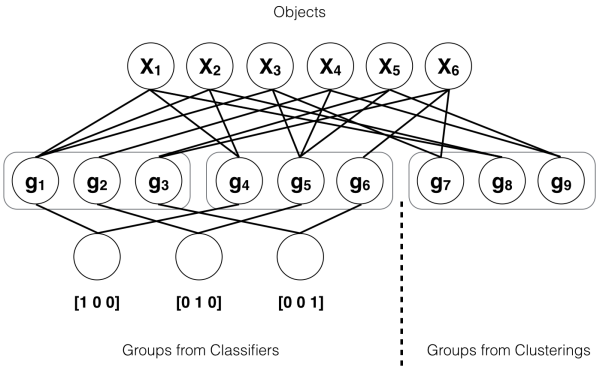


Figure 1: Bipartite Graph Representation

Table 1: Notations

Symbol	Definition
$A_{n\times v} = [a_{ij}]$	indicator of object i in group j
$U_{n\times c} = [u_{ij}]$	probability of object i w.r.t. class j
$Q_{v\times c} = [q_{ij}]$	probability of group i w.r.t. class j
$Y_{v\times c} = [y_{ij}]$	indicator of object i predicted as class j

2 Consensus-based prediction combination

The goal of consensus-based prediction combination is to reach a maximal consensus among multiple models using the predictions from multiple base models [14]. Supposed there are m base models, in which r models are classifiers and $m-r$ are clustering models, each model will give a label (or group ID) for each data example in the data set. Assume that the number of classes is c, m models will give m predictions partition the data set into mc groups, in which rc groups are classes produced by supervised models, and $(m-r)c$ groups are clusters got from unsupervised models. Therefore, n data objects in the data set and mc group nodes can form a bipartite graph. Figure 1 shows the bipartite graph representation of consensus-based prediction combination. We denote the connections in the bipartite graph representation by a series of matrices. Table 1 gives the detailed notations.

Suppose the total number of groups is $v = mc$, we use a matrix $A_{n\times v}$ to denote the relationship between data objects and groups. We set $a_{ij} = 1$ if object i is classified or clustered into group j by a model, 0 oth-

erwise. We use another matrix $U_{n\times c}$ to represent class assignments given by the model combination method. Similarly, Q_{vc} denotes the class memberships of the v group nodes. $Y_{v\times c}$ denotes the corresponding relations among groups and classes, where $y_{ij} = 1$ if the i-th group node belongs to class j and 0 otherwise.

Then we formulate the consensus-based combination problem by an objective function (Let $k_j = \sum_1^c y_{iz}$):

$$\begin{aligned} \min_{Q,U} \quad & \sum_{i=1}^{n}\sum_{j=1}^{v} a_{ij}\|u_{i.} - q_{j.}\|^2 + \alpha \sum_{j=1}^{v} k_j \|q_{j.} - y_{j.}\|^2 \\ \text{s.t.} \quad & u_{i.} \geq 0, \quad |u_{i.}| = 1, \quad i = 1, \ldots, n \\ & q_{j.} \geq 0, \quad |q_{j.}| = 1, \quad j = 1, \ldots, v \end{aligned}$$

Note that in Eq.(2.1), the first term ensures the constraint that objects and groups which are connected in the graph are possible to have close labels. And the second term imposes the constraint that the group label estimate should not deviate much from its initial class label prediction. This optimization problem [14] can be solved using block coordinate descent methods, iteratively updating matrix U and Q until U converges.

3 Online Consensus Maximization

Consensus maximization classification is very when combining data from multiple sources using multiple base models. Nowadays, web technology allows every people to be a "sensor" in social network, therefore, the sources providing information are heterogeneous and there come different kinds of models to deal with data in different categories. If we want to give a specific user a label on his or her interest based on the profile, we may need to combine all contents in the profile together to give the label. For example, the textual blogs, photos and web links should be treated differently in different models. Another example is an application in network security. Hackers will always change their behaviors, so that it is hard to build a single model to detect an attack effectively. In contrast, what we should do is to build multiple models with respect to different major features, and combine the results together to decide if an action is an attack.

In both the applications above, it is obvious that the task cannot be done after all data are received. Social behaviors and network actions will form endless data streams, and immediate response is required at each arrival of new data. Because prediction on social network should be real-time, or the advertisement/recommendation will lose its value, and it is more important to keep the network safe from any possible attack every second. Online consensus maximization is a valuable problem, which can provide a great improvement to a wide range of applications similar to the two examples.

More formally, suppose we have a data stream, denoted by S. Each instance in S contains the output labels from r classification models and $(m-r)$ clustering algorithms on a dataset X. The task of online consensus maximization is, give a predictive label to each arriving data example, based on the m labels produced by base models, and the information maintained by the model retrieved from previous data examples. Figure 2 shows the process of online consensus maximization, different from the traditional consensus maximization.

In traditional consensus maximization model, base models collect raw data and produce labels as the input to CM model. Then CM model iteratively computes the maximized consensus labels as the final predictions. Information flows are shown by arrows. However, things change in the data stream case. The raw data cannot be collected in advance, instead, there is a data stream which need to be keep track with in real time. Base models may not be synchronized well, as the dash arrow describes, some models will give late response. Furthermore, OnlineCM model will keep updating when new data objects come into it.

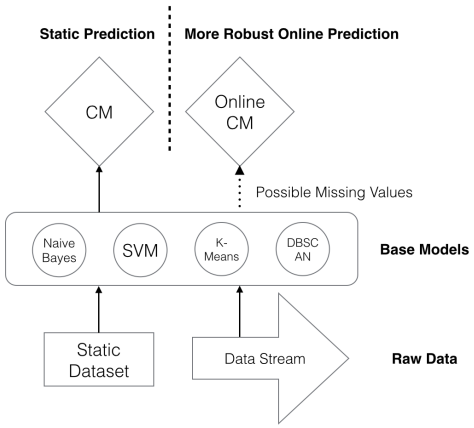


Figure 2: Comparison between Traditional and Online Consensus Maximization

4 Algorithm

Consensus maximization essentially iterates between the following two equations until it converges.

$$Q^{(t)} = (D_v + \alpha K_v)^{-1}(A'U^{(t-1)} + \alpha K_v Y) \tag{4.1}$$

$$U^{(t)} = D_n^{-1} A Q^{(t)} \tag{4.2}$$

The iterative nature of CM has the following drawbacks in an online learning setting:

- The input to CM, namely, A needs to be known before running the algorithm. This assumption is not true in an online setting, where the predictions of instances keep streaming in and A is not a constant, the above equations are not no longer applicable.

- The size of memory necessary to run the algorithm grows with the size of input. This is undesirable in an online setting, where the size of input can be unbounded.

- Multiple scans of the data is needed to produce the output, while in an online setting, predictions are usually produced in real-time.

In the following sections, we describe an online prediction aggregation algorithm to handle the above challenges.

4.1 Online Fast Prediction The key to transform the iterative batch CM into an any-time online algorithm, we re-formulate the iterative update equations as closed form solutions, which enable online computing without keeping all input.

First we give some useful notations:

- D_v: D_v is diagonal with the j-th diagonal elements being $\sum_{i=1}^{n} a_{ij}$, as we add one more row to A, D_v is updated by $\sum_{i=1}^{n} a_{ij} + a_{n+1,j}$.

- D_λ: $D_\lambda = (D_v + \alpha K_v)^{-1} D_v$, therefore, once we updated D_v, D_λ can be easily derived from D_v.

- $D_{1-\lambda}$: $D_{1-\lambda} = (D_v + \alpha K_v)^{-1}(\alpha K_v)$, which is similar to D_λ.

- $S = D_v^{-1} A' D_n^{-1} A$: since we know how to update D_v, we just need to update $A' D_n^{-1} A$. The new ij-th entry of $A' D_n^{-1} A$ is given by $\frac{1}{m}\sum_{k=1}^{n+1} a_{ik} a_{kj} = \frac{1}{m}(\sum_{k=1}^{n} a_{ik} a_{kj} + a_{i,n+1} a_{n+1,j})$.

As what has been done in [14], plugging Eq.(4.2) into Eq.(4.1) gives

$$Q^{(t)} = (D_v + \alpha K_v)^{-1}(A' D_n^{-1} A Q^{(t-1)} + \alpha K_v Y) \tag{4.3}$$

As $t \to \infty$, $Q^{(t)}$ converges to

$$Q^* = (I - D_\lambda S)^{-1} D_{1-\lambda} Y \tag{4.4}$$

where $S = D_v^{-1} A' D_n^{-1} A$, which can be seen as a similarity matrix for group nodes. Different from [14] that uses another closed form formula to obtain the consensus predictions U, we can compute U using Eq.(4.2), with Q being replaced with Q^*. Specifically,

$$U = D_n^{-1} A Q^* = \frac{1}{m} A Q^* \tag{4.5}$$

Eq.(4.4) and Eq.(4.5) fulfill the goals we discussed in Section 4. First, the space requirement is kept minimal. Only the matrix S needs to be maintained and its size is of $O(m^2 c^2)$, which has nothing to do with the size of the data. Second, consensus predictions can be produced online, without waiting for all consensus predictions to be ready. In Eq.(4.5), given Q^* captures all necessary information, prediction of the last instance is given by

$$u_{n+1,\cdot} = \frac{1}{m} \sum_{j=1}^{v} a_{n+1,j} Q^*_{j\cdot} \tag{4.6}$$

Lastly and most importantly, the model can be updated as predictions for instances arrive one by one. To see this, we list the variables that need to be updated when the predictions of a new instance come in. After we update D_v, D_λ, $D_{1-\lambda}$ and S, it is straightforward to obtain the consensus prediction for the current instance using Eq.(4.5). One might have noted that the model training is decoupled from consensus prediction, while the methods in [14, 13] couple the training and prediction phases.

4.2 Missing Values Imputation The above described algorithm assumes that there is no missing values in base model predictions, and the matrix A and S can be updated with full information. However, as we pointed out, base models can be out of synchronization, or data from some of the sources can be missing, leading to missing values in the matrix A. In the experiments, we show that such missing predictions can greatly affect the performance of the above-mentioned online algorithm. Therefore we need to handle the missing predictions, and imputation is a effective way for such purpose. For example, a missing value in the feature vector can be replaced with the mean or median of the values of that feature in the observed data. More complicated imputation methods, such as model-based imputation have also been proposed and shown to be powerful. The challenge here is that such imputation should not impose significant overhead in running time and space, and should be able to work in the streaming environment.

Here we adopt a regression based approach to handle the missing predictions. For simplicity, we assume that there is at most one model will miss its prediction for any instance. We also assume that there are sufficient amount of instances that all models are able to

give their predictions (called "full predictions"). The approach learns the dependency between the missing and non-missing predictions from full predictions, and the dependency is used to impute the missing prediction given the non-missing ones for the same instance. More formally, given a set of full predictions from m base models, m regression models can be trained, with the k-th model taking the k-th entry in the prediction vector as response, and the remaining entries as variates/features. Then given that an incomplete prediction vector with the k-th entry missing, the missing entry can be imputed using the k-th regression model. This schema is shown in Figure 3. In our experiments, we use softmax regression, which is a generalization of logistic regression to multi-class problems.

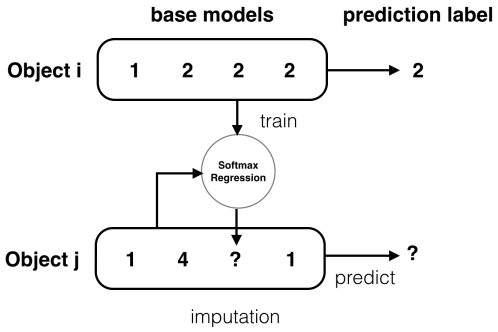


Figure 3: Softmax regression imputation model

5 Experiments

We evaluate our proposed model from two perspectives. First, we compare the prediction performance of the proposed online consensus maximization algorithm with that of two baselines, namaly, the batch mode and incremental consensus maximization. Second, we provide a sensitivity analysis of the proposed algorithm. Specifically, we vary the percentage of of missing values in the base models' predictions and show how proposed softmax regression based imputation method can handle missing values in data stream effectively and efficiently.

5.1 Data Sets Table 2 gives the statistics of the four datasets employed for evaluation. For each dataset, two supervised models and 2 unsupervised models serve as base models, each of which predicts a label or cluster id for each testing example. These predictions of base models serve as the input to the proposed algorithm and the baselines.

5.2 Baselines BGCM[14]: on the one hand, BGCM is only applicable in batch mode, which requires the availability of all base models' predictions. On the other hand, although BGCM can still give reasonable aggregation of base models in the presence of missing values, it does not specify a principled way to impute missing values. Since the batch mode method has access to all available information, its performance is expected to be the upper bound of any online variations of BGCM. By comparing with BGCM, we can see how far away the performance of the proposed method is from the batch mode.

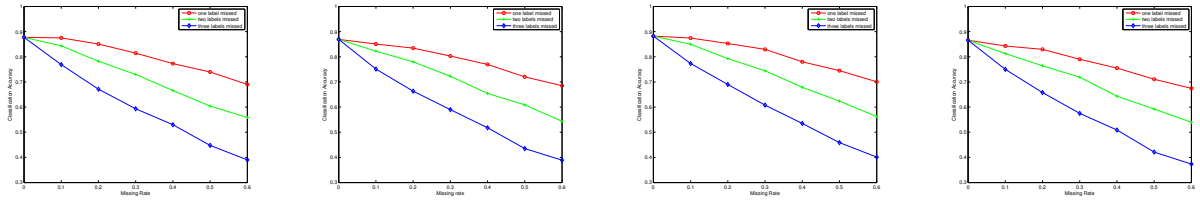
Table 2: Data Sets Description

Data Set	# of training	# test	# classes	# features
rcv1	15564	518571	53	47236
mnist	60000	10000	10	778
SensIT	25010	1000000	10	10
covtype	20000	561012	7	54

IncrementalCM[13]: this method is an extension of BGCM for prediction aggregation in an incremental manner, though its update rules are quite different from those we proposed in this paper. Briefly speaking, given the base models' predictions of the current instance, say $\mathbf{x}_n$, its posterior class distribution, denoted by $\mathbf{u}_n$, is first obtained by averaging the class distributions of the group nodes that this instance connects to. Then $\mathbf{u}_n$ is used to update the class distributions of grounp nodes as in BGCM. Lastly, all previous instances' posterior class distributions are updated. In a streaming environment, the updates in the last step is infeasible and we simply ignore these updates. By comparing with IncrementalCM, we show that the proposed method's online update rule is more appropriate in capturing model correlation. Note that IncrementalCM does not specify how to handle missing values either.

5.3 Online Consensus Maximization on Complete Datasets Table 3 shows the accuracy of the aggregation predictions based on the proposed method and the baselines. The accuracy is computed over all instances' aggregated predictions. From the table, we can observed that the proposed method significantly outperforms IncrementalCM overall all datasets. For example, the improvement is over 6% on the rcv1 dataset, and 6.5% on the covtype dataset. Another interesting observation is that even the proposed algorithm works in an online fashion, it does suffer too much from the unavailability of the full set of predictions, and the OnlineCM's performance is quite close to the upper bound of performance given by BGCM.

5.4 Sensitivity to Missing Rate In big data applications, sometimes the prediction speed of some base models may not be able to keep up with the speed in



(a) rcv1 (b) ensIT (c) mnist (d) covtype

Figure 4: Sensitivity of the proposed algorithm to missing predictions

Table 3: Classification Accuracy

Methods	rcv1	mnist	SensIT	covtype
BGCM	0.8954	0.9187	0.8843	0.8901
OnlineCM	**0.8772**	**0.8822**	**0.8692**	**0.8758**
IncrementalCM	0.8240	0.8415	0.8375	0.8176

which data is generated. Therefore, some base models may choose to skip part of the data when there is a lag, leading to missing values in the stream of predictions. Such absence of predictions poses a challenge for almost all predictions combination methods. We create predictions with missing values to study the sensitivity of OnlineCM without missing value imputation. Two parameters control how much base models will miss predictions. The first parameter is the ratio of instances that will contain missing values. We vary this parameter from 0 to 60%, that is, from "no missing value" to 60% percent of the instances will contain missing predictions. The second parameter is how many models will miss their predictions for an instance. Since we have 4 base models, we let this parameter take values from 1, 2 and 3. The sensitivity of the proposed algorithm to different levels of missing value is shown in Figures 4(a) to 4(d). As expected, the more instances that contain missing predictions, or the more base models miss their predictions, the accuracy of the aggregated predictions degrades severely, and therefore a principled treatment of the missing values is needed.

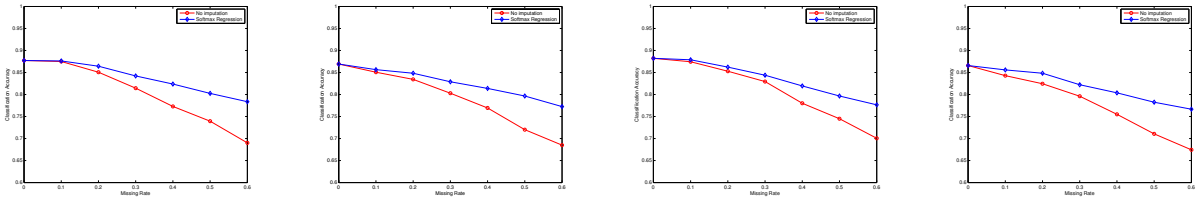
5.5 Effectiveness of missing value imputation

In this part, we will show the effectiveness of our proposed algorithm on solving the missing value problem. We vary the percentage of instances that contain missing predictions from 0 to 60%, but only on base models is randomly picked from the 4 base models to have a missing prediction for each instance that is sampled to have missing predictions. Figure 5 compares the accuracy of the proposed approach with and without missing value imputation. We can see that the softmax regression based method can significantly mitigate the damages caused by missing predictions, especially when the number of instances that have missing predictions is high. For example, in Figure 5(d), when there are 60% of instances containing missing predictions, the proposed imputation method can save about 10% in accuracy. The explanation of such improvement is that, as the regression model sees more and more base models' predictions, the relationship between models can be learned more and more accurately. Therefore, the missing predictions can be accurately predicted using the regression model based on past predictions. From the curves above, we can get the knowledge of that, with the increase of missing rate, the classification accuracy decreases a lot in traditional method where no imputation is applied. Our approach enhances the accuracy under the same level of missing rate, and slows down the decrease in accuracy to a certain extent, which makes consensus maximization model more applicable in real world complex scenarios.

With the same imputation method when missing values exist, it is straightforward to see IncrementalCM cannot have a better performance than OnlineCM, since it has an even lower accuracy on complete data sets.

6 Related Works

Ensemble methods [3, 9] have been an important and powerful family in machine learning and data mining. Classical ensemble methods focus on training an ensemble of models to improve classification or clustering performance over single base models. For example, bagging [4] uses bootstrap subsample of the original training data to training multiple diverse base classifiers, such as decision trees [6] to improve classification accuracy. Boosting [5] is another popular ensemble training method, where weak classifiers are trained in a sequential manner and the current classifier depends on previous model to focus on difficult instances in the data. The improvement of boosting can be explained using bias-variance reduction [5] or large margin learning [25]. For an overview of ensemble methods, please refer to the excellent textbooks [34, 16].



(a) rcv1 (b) ensIT (c) mnist (d) covtype

Figure 5: Imputation Effectiveness on 4 data sets

Model combination methods focus on combining the output of base models to arrive at consolidated predictions, without access to the data or models. Roughly speaking, these methods can be categorized into the following three families based on the techniques they use.

Matrix factorization In [19, 20], the authors proposed to treat the predictions of base models as a matrix, which is then factorized using symmetric non-negative matrix factorization. The resulting factor matrix is used as clustering indicators to group the instances into clusters. In [33], the authors proposed to use robust matrix completion to aggregate multiple clustering results. These methods focus on combining unsupervised models,

Probabilistic approaches These methods treat the prediction matrix as observed data, while model the instance memberships as latent variables. In [22], they proposed a probabilistic embedding method to combine multiple supervised and unsupervised models. In [29], they proposed an LDA-like Bayesian hierarchical model to model the latent variables. In [2], they extended [29] to transfer learning setting, and in [1] to non-transductive setting. In [30], a nonparametric Bayesian method is proposed to infer the number of clusters automatically from the predictions. Usually, these probabilistic approaches require more computational time on sampling or variational inference.

Graph based approaches These methods models the relationships between instances using graphs, from which aggregated predictions can be inferred. The pioneering work [27] introduced three graph based algorithms to aggregate base model predictions. For example, HGPA in [27] constructs a hypergraph consisting of membership indicators from clustering models as hyperedges, then a hypergraph partition algorithm partitions the hypergraph. Another method in [27], MCLA, partitions the hypergraph into k subgraphs, each of which consists of membership indicators from clustering models. In [14], they proposed BGCM to infer instance memberships from a bipartite graph constructed from base model predictions. In [11], they build a similar bipartite graph, which is partitioned using spectral clustering [24] or METIS [17].

Recent development of model combination In [31], they proposed to jointly learn the weights of the base models and infer the ground truths in an iterative way. The idea is that the base model weights and ground truths can help each other, leading to better results. In [32] they studied the overfitting problem in model combination, and proposed a regularization framework to handle the issue.

Missing values arise in practical applications. For example, people might choose to ignore some questions in a survey; in medicine, a patient may not have the results of certain examinations. How to deal with missing values to improve the performance of data analysis models has long been studied and imputation is one popular method [15]. A simple missing value imputation approach is to fill up the missing values of a feature using the median or mean of that feature. The short coming of this approach is that different instances should have different values even for the same feature. Model-based missing value imputation [26] considers such dependency of the missing values on the instances. Models are built to capture the relationships between features, and missing values are predicted using the non-missing values of the same instance. Recently, matrix or tensor completion are employed to fill up the missing values in matrices or tensors [23, 21].

Learning from data streams is highly desirable for application dealing with big data [10], and there are several tasks in learning from streams. Online learning usually refers to learning a classifier in an incremental way [7]. Online clustering assigns cluster ids to instances that coming in a stream, while the underlying clustering model is updated using the current instance [8]. Handling non-stationary distribution in the data streams is an important research problem [28, 12, 18].

7 Conclusion

In this paper, we present an online consensus maximization method for classification problem with missing data, one of the ensemble classification methods which can improve accuracy and robustness over a single model in data streams. It gives predictions based on the labels generated by a series of basic models, supervised and unsupervised, gains good performance without access to raw data.This is the first work trying to build a consensus ensemble classification model in data streams, considering the possible missing values as well. Based on our experimental results, we find our proposed method can improve the accuracy significantly, and compete over the baselines. On the one hand, OnlineCM solves the problem that traditional consensus maximization (BGCM) cannot work in data streams, which is the advantage in efficiency. On the other hand, OnlineCM beats the IncrementalCM in effectiveness. Furthermore, it provides a solution to possible missing values in real world application. The main contributions are practical and important in the era of big data.

Some future extensions are possible. One option is to combine multiple related data stream to improve both of their prediction, which is also referred as transfer learning. The other interesting issue is how to use the "late" labels of some basic models to improve the previous "temporary" prediction result, which may improve the present result even better.

Acknowledgements

This work is supported in part by NSF grants (CNS-1115234, OISE-1129076, and IIS-1319973) and Huawei grant.

References

[1] Acharya Ayan, Hruschka Eduardo, R., Ghosh Joydeep, Sarwar Badrul, and Ruvini Jean-David. Probabilistic combination of classifier and cluster ensembles for non-transductive learning. In *SDM*, 2013.

[2] Acharya Ayan, R. Hruschka Eduardo, Ghosh Joydeep, and Acharyya Sreangsu. An optimization framework for semi-supervised and transfer learning using multiple classifiers and clusterers. In *CoRR*, 2012.

[3] Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine learning*, 36(1-2):105–139, 1999.

[4] Leo Breiman. Bagging predictors. *Mach. Learn.*, 24(2), 1996.

[5] Leo Breiman. Bias, variance, and arcing classifiers. Technical report, 1996.

[6] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[7] Nicolò Cesa-Bianchi, Alex Conconi, and Claudio Gentile. A second-order perceptron algorithm. *SIAM J. Comput.*, 34(3):640–668, 2005.

[8] Anna Choromanska and Claire Monteleoni. Online clustering with experts. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, 2012.

[9] Thomas G Dietterich. Ensemble methods in machine learning. In *Multiple classifier systems*, pages 1–15. Springer, 2000.

[10] Pedro Domingos and Geoff Hulten. Mining high-speed data streams. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 71–80. ACM, 2000.

[11] Xiaoli Zhang Fern and Carla E. Brodley. Solving cluster ensemble problems by bipartite graph partitioning. ICML, 2004.

[12] Venkatesh Ganti, Johannes Gehrke, and Raghu Ramakrishnan. Mining data streams under block evolution. *ACM SIGKDD Explorations Newsletter*, 3(2):1–10, 2002.

[13] Jing Gao, Wei Fan, Deepak Turaga, Olivier Verscheure, Xiaoqiao Meng, Lu Su, and Jiawei Han. Consensus extraction from heterogeneous detectors to improve performance over network traffic anomaly detection. In *INFOCOM*, 2011.

[14] Jing Gao, Feng Liang, Wei Fan, Yizhou Sun, and Jiawei Han. Graph-based consensus maximization among multiple supervised and unsupervised models. In *NIPS*, 2009.

[15] G. D. Garson. *Missing Values Analysis and Data Imputation*. Statistical Associates Publishers, 2012.

[16] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition, 2009.

[17] George Karypis and Vipin Kumar. Metis – unstructured graph partitioning and sparse matrix ordering system, version 2.0. Technical report, 1995.

[18] Mark Last. Online classification of nonstationary data streams. *Intelligent Data Analysis*, 6(2):129–147, 2002.

[19] Tao Li and Chris Ding. Weighted Consensus Clustering. SDM, 2008.

[20] Tao Li, Chris Ding, and Michael I. Jordan. Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization. ICDM, 2007.

[21] Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor completion for estimating missing values in visual data. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(1), January 2013.

[22] Xudong Ma, Ping Luo, Fuzhen Zhuang, Qing He, Zhongzhi Shi, and Zhiyong Shen. Combining supervised and unsupervised models via unconstrained probabilistic embedding. IJCAI, 2011.

[23] Sahand Negahban and Martin J. Wainwright. Restricted strong convexity and weighted matrix completion: Optimal bounds with noise. *J. Mach. Learn. Res.*, 13, May 2012.

[24] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss.

On spectral clustering: Analysis and an algorithm. In *NIPS*, 2001.

[25] G. Rtsch, T. Onoda, and K.-R. Mller. Soft margins for adaboost. *Machine Learning*, 42(3):287–320, 2001.

[26] Maytal Saar-Tsechansky and Foster Provost. Handling missing values when applying classification models. *J. Mach. Learn. Res.*, 2007.

[27] Alexander Strehl and Joydeep Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 2003.

[28] Haixun Wang, Wei Fan, Philip S Yu, and Jiawei Han. Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 226–235. ACM, 2003.

[29] Hongjun Wang, Hanhuai Shan, and Arindam Banerjee. Bayesian cluster ensembles. In *SDM*, 2009.

[30] Pu Wang, Carlotta Domeniconi, and Kathryn Blackmond Laskey. Nonparametric bayesian clustering ensembles. In *ECML PKDD*, 2010.

[31] Sihong Xie, Wei Fan, and Philip S. Yu. An iterative and re-weighting framework for rejection and uncertainty resolution in crowdsourcing. In *SDM*, 2012.

[32] Sihong Xie, Jing Gao, Wei Fan, Deepak Turaga, and Philip S. Yu. Class-distribution regularized consensus maximization for alleviating overfitting in model combination. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 303–312, 2014.

[33] Jinfeng Yi, Tianbao Yang, Rong Jin, A.K. Jain, and M. Mahdavi. Robust ensemble clustering by matrix completion. ICDM, 2012.

[34] Zhi-Hua Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, 1st edition, 2012.