Classification of Gene Expression Data: A Hubness-aware Semi-supervised Approach

Krisztian Buza
Brain Imaging Center
Research Center for Natural Sciences
Hungarian Academy of Sciences, Budapest, Hungary
buza@biointelligence.hu
http://www.biointelligence.hu

Abstract

Background and Objective Classification of gene expression data is the common denominator of various biomedical recognition tasks. However, obtaining class labels for large training samples may be difficult or even impossible in many cases. Therefore, semi-supervised classification techniques are required as semi-supervised classifiers take advantage of unlabeled data.

Methods Gene expression data is high-dimensional which gives rise to the phenomena known under the umbrella of the *curse of dimensionality*, one of its recently explored aspects being the presence of hubs or *hubness* for short. Therefore, hubness-aware classifiers have been developed recently, such as Naive Hubness-Bayesian k-Nearest Neighbor (NHBNN). In this paper, we propose a semi-supervised extension of NHBNN which follows the self-training schema. As one of the core components of self-training is the certainty score, we propose a new hubness-aware certainty score.

Results We performed experiments on publicly available gene expression data. These experiments show that the proposed classifier outperforms its competitors. We investigated the impact of each of the compo-

nents (classification algorithm, semi-supervised technique, hubness-aware certainty score) separately and showed that each of these components are relevant to the performance of the proposed approach.

Conclusions Our results imply that our approach may increase classification accuracy and reduce computational costs (i.e., runtime). Based on the promising results presented in the paper, we envision that hubness-aware techniques will be used in various other biomedical machine learning tasks. In order to accelerate this process, we made an implementation of hubness-aware machine learning techniques publicly available in the PyHubs software package (http://www.biointelligence.hu/pyhubs) implemented in Python, one of the most popular programming languages of data science.

Keywords Gene expression, machine learning, semi-supervised classification, high dimensionality.

1 Introduction

Various tissues are characterized by different gene expression patterns. Additionally, a number of diseases and disease subtypes may be associated with characteristic gene expression patterns. Therefore, recognition tasks related to gene expression data may contribute to the diagnosis of various diseases such as colon cancer, lymphoma, lung cancer and subtypes of breast cancer [9]. Due to the large amount of data (e.g., even if we consider just a single patient, expression levels of thousands of genes may be measured), such recognition tasks are typically solved by computers, and state-of-the-art solutions are based on machine learning.

In case of supervised machine learning, a previously collected dataset (e.g., gene expression levels measured for a set of patients) together with evidence or indication (e.g., the presence, absence or subtype of a particular disease for

each patient) is used to induce a decision model, called *classifier*. Once the classifier is induced, it will be able to solve the recognition task for new data instances (e.g., the classifier will be able to recognize the subtype of cancer for new patients). With *training the classifier* we refer to the induction of the model, while the data used to induce the model is called *training data*. If the data is associated with evidence, it is called *labeled data*, e.g., a labeled dataset may contain gene expression levels together with the information describing which patient has which subtype of cancer, in contrast, if only the gene expression levels are available without knowing the subtype or presence of the disease, the dataset is unlabeled. The value of the evidence is called *label*, e.g., if a patient has estrogen receptor positive (ER+) subtype of breast cancer, we say its label is "ER+" (at the technical level, labels are usually coded by integer numbers, such as 0 for "ER+" and 1 for "ER−").

The classification task is challenging for several reasons. Usually, the expression levels of several thousands of genes are measured, therefore, the data is high-dimensional which gives rise to the phenomena known under the umbrella of the *curse of dimensionality* [3]. While well-studied aspects of the curse are the sparsity and distance concentration, see e.g. [19], a recently explored aspect of the curse is the presence of hubs [14], i.e., instances that are similar to surprisingly many other instances. According to recent observations, the presence of hubs characterizes gene expression datasets [10],[15]. A hub is said to be bad if its class label differs from the class labels of those instances that have this hub as one of their k-nearest neighbors. In the context of k-nearest neighbor classification, bad hubs were shown to be responsible for a surprisingly large portion of the total classification error.

Recently, algorithms have been developed under the umbrella of hubness-aware data mining, see e.g. [5],[12],[13],[15],[22],[25],[26],[20] and [21] for a sur-

vey. These algorithms try to recognize bad hubs and reduce their influence on classifications of unlabeled instances.

It may be expensive (or even impossible in case of rare diseases) to collect *large* amount of labeled gene expression data, therefore, we have to account for the fact that only relatively few labeled instances are available which may not reflect the structure of the classes well enough. Therefore, while training the classifier, in addition to learning from labeled data, the classifier should be able to use unlabeled data too in order to discover the structure of the classes.

In this paper we introduce a semi-supervised hubness-aware classifier, i.e., a classifier that uses both labeled and unlabeled data for training. In particular, our approach is an extension of the Naive Hubness Bayesian k-Nearest Neighbor, or NHBNN for short [23], which is one of the most promising hubness-aware classifiers. As we will show, straightforward incorporation of semi-supervised classification techniques with NHBNN leads to suboptimal results, therefore, we develop a hubness-aware inductive semi-supervised classification scheme. We propose to use our classifier for recognition tasks related to gene expression data. To our best knowledge, this paper is the first that studies hubness-aware semi-supervised classification of gene expression data.

2 Methods

Semi-supervised classification, often in a general data mining context, i.e., without special focus on the analysis of genetic data, has been studied intensively, see e.g. [6],[11] and the references therein for related works on semi-supervised classification. In order to ensure that our study is self-contained, we begin this section by reviewing the Naive Hubness Bayesian k-Nearest Neighbor (NHBNN) classifier [23] and the self-training semi-supervised learning technique in Section 2.1 and Section 2.2. The presentation of NHBNN and self-training is based

on [21] and [11] respectively. Subsequently, we describe our proposed semi-supervised approach in Section 2.3, which is followed by the methods used for the experimental evaluation in Section 2.4.

2.1 NHBNN: Naive Hubness Bayesian k-Nearest Neighbor

Notation

$\mathcal{N}_k(x)$ denotes the set of k-nearest neighbors of x.

$P(y = C|\mathcal{N}_k(x))$ denotes probability that x belongs to class C given its nearest neighbors.

$P(x \in \mathcal{N}_k|C)$ denotes the probability of the event that x appears as one of the k-nearest neighbors of any labeled training instance belonging to class C.

$P(C)$ denotes the prior probability of the event that an instance belongs to class C.

We aim at classifying instance x^*, i.e., we want to determine its unknown class label y^*. We use $\mathcal{N}_k(x^*)$ to denote the set of k-nearest neighbors of x^*. For each class C, Naive Hubness Bayesian k-Nearest Neighbor (NHBNN) estimates $P(y^* = C|\mathcal{N}_k(x^*))$, i.e., the probability that x^* belongs to class C given its nearest neighbors. Subsequently, NHBNN selects the class with highest probability.

NHBNN follows a Bayesian approach to assess $P(y^* = C|\mathcal{N}_k(x^*))$. For each labeled training instance x, one can estimate the probability of the event that x appears as one of the k-nearest neighbors of any labeled training instance belonging to class C. This probability is denoted by $P(x \in \mathcal{N}_k|C)$. While calculating nearest neighbors, throughout this paper, an instance x is *never* treated as the nearest neighbor of itself, i.e., $x \notin \mathcal{N}_k(x)$.

Assuming conditional independence between the nearest neighbors given the class, $P(y^* = C|\mathcal{N}_k(x^*))$ can be assessed as follows:

$$P(y^* = C|\mathcal{N}_k(x^*)) \quad \propto \quad P(C) \prod_{x_i \in \mathcal{N}_k(x^*)} P(x_i \in \mathcal{N}_k|C). \tag{1}$$

where $P(C)$ denotes the prior probability of the event that an instance belongs to class C. From the labeled training data, $P(C)$ can be estimated as

$$P(C) \approx \frac{|\mathcal{D}_C^{lab}|}{|\mathcal{D}^{lab}|}, \tag{2}$$

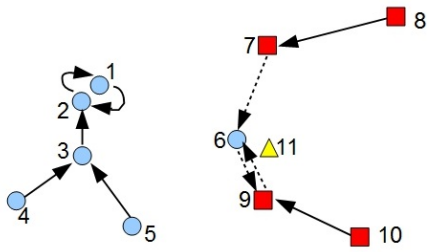

Figure 1: Running example used to illustrate NHBNN. Labeled training instances belong to two classes, denoted by circles and rectangles. From each labeled training instance, a directed edge points to its first nearest neighbor among the labeled training instances. The triangle is an instance to be classified. For details, see the description of NHBNN.

where $|\mathcal{D}_C^{lab}|$ denotes the number of labeled training instances belonging to class C and $|\mathcal{D}^{lab}|$ is the total number of labeled training instances. The maximum likelihood estimate of $P(x_i \in \mathcal{N}_k|C)$ is the fraction

$$P(x_i \in \mathcal{N}_k|C) \approx \frac{N_{k,C}(x_i)}{|\mathcal{D}_C^{lab}|}, \tag{3}$$

where $N_{k,C}(x_i)$ denotes the (k, C)-occurrence of an instance x_i, i.e., how many times x_i occurs as one of the k-nearest neighbors of labeled training instances belonging to class C.

Notation (cont.)

$N_{k,C}(x)$ denotes how many times x occurs as one of the k-nearest neighbors of labeled training instances belonging to class C.

Example. Fig. 1 shows a simple two-dimensional example, i.e., instances, denoted from now on as $x_1, \ldots, x_{11}$ in text, correspond to points of the plane. In this example, we use $k = 1$. In Fig. 1, a directed edge points from each labeled training instance to its first nearest neighbor among the labeled training instances. In other words: the nearest neighbor relationships shown in the Fig. 1 are calculated solely on the *labeled training data*.

Out of the ten labeled training instances, six belong to the class of circles (C_1) and four belong to the class of rectangles (C_2). Thus: $|\mathcal{D}_{C_1}^{lab}| = 6$, $|\mathcal{D}_{C_2}^{lab}| = 4$,

$P(C_1) = 0.6$ and $P(C_2) = 0.4$. Next, we calculate $N_{k,C}(x_i)$ for both classes and classify x_{11} using its first nearest neighbor, i.e., x_6. In particular, Eq. (3) leads to

$$P(x_6 \in \mathcal{N}_1|C_1) \approx \frac{N_{1,C_1}(x_6)}{|\mathcal{D}_{C_1}^{lab}|} = \frac{0}{6} = 0$$

and

$$P(x_6 \in \mathcal{N}_1|C_2) \approx \frac{N_{1,C_2}(x_6)}{|\mathcal{D}_{C_2}^{lab}|} = \frac{2}{4} = 0.5.$$

According to Eq. (1) we calculate

$$P(y_{11} = C_1|\mathcal{N}_2(x_{11})) \propto 0.6 \times 0 = 0$$

and

$$P(y_{11} = C_2|\mathcal{N}_2(x_{11})) \propto 0.4 \times 0.5 = 0.2.$$

As $P(y_{11} = C_2|\mathcal{N}_2(x_{11})) > P(y_{11} = C_1|\mathcal{N}_2(x_{11}))$, x_{11} will be classified as a rectangle.

The previous example also illustrates that estimating $P(x_i \in \mathcal{N}_k|C)$ according to (3) may simply lead to zero probabilities. In order to avoid this, we can use a simple Laplace-estimate for $P(x_i \in \mathcal{N}_k|C)$ as follows:

$$P(x_i \in \mathcal{N}_k|C) \approx \frac{N_{k,C}(x_i) + m}{|\mathcal{D}_C^{lab}| + mq}, \tag{4}$$

where $m > 0$ and q denotes the number of classes. Informally, this estimate can be interpreted as follows: we consider m additional pseudo-instances from each class and we assume that x_i appears as one of the k-nearest neighbors of the pseudo-instances from class C. We use $m = 1$ in our experiments.

Even though (k, C)-occurrences are highly correlated, as shown in [21] and [23], NHBNN offers improvement over the basic kNN. This is in accordance with

other results from the literature that state that Naive Bayes can deliver good results even in cases with high independence assumption violation [16].

2.2 Self-training

Self-training is one of the most commonly used semi-supervised algorithms. Self-training is a wrapper method around a supervised classifier, i.e., one may use self-training to enhance various classifiers. To apply self-training, for each instance x^* to be classified, besides its predicted class label, the classifier must be able to output a certainty score, i.e., an estimation of how likely the predicted class label is correct.

Self-training is an iterative process during which the set of labeled instances is grown until all the instances become labeled. Let L_t denote the set of labeled instances in the t-th iteration ($t \geq 0$) while U_t shall denote the set of unlabeled instances in the t-th iteration. L_0 denotes the instances that are labeled initially, i.e., the labeled training data, while U_0 denotes the set of initially unlabeled instances. In each iteration of self-training, the base classifier is trained on the labeled set L_t. Then, the base classifier is used to classify the unlabeled instances. Finally, the instance with highest certainty score is selected. This instance, together with its predicted label $\hat{y}$, is added to the set of labeled instances in order to construct L_{t+1} the set of labeled instances in the next iteration. We refer to [11] for the pseudocode and an illustration of the self-training algorithm.

If an unlabeled instance is classified incorrectly and this instance is added to the training data of the subsequent iterations, this may cause a chain of classification errors. Therefore, as noted in [8], it may be worth to stop self-training after a moderate number of iterations and use the resulting model to label all the remaining unlabeled instances.

2.3 Certainty Estimation for NHBNN

In order to allow NHBNN to be used in self-training mode, we only need to define an appropriate certainty score. A straightforward certainty score may be based on the probability estimates as follows:

$$certainty(x^*) = \frac{P(C') \prod\limits_{x_i \in \mathcal{N}_k(x^*)} P(x_i \in \mathcal{N}_k | C')}{\sum\limits_{C_j \in \mathcal{C}} \Big(P(C_j) \prod\limits_{x_i \in \mathcal{N}_k(x^*)} P(x_i \in \mathcal{N}_k | C_j) \Big)}. \tag{5}$$

where C' denotes the class with maximal estimated probability and $\mathcal{C}$ denotes the set of all the classes. In the example shown in Fig. 1, the above certainty estimate gives

$$\frac{0.2}{0 + 0.2} = 1$$

when classifying x_{11}.

However, this certainty estimate does not take into account that, usually, unlabeled instances appearing as nearest neighbors of many labeled instances can be classified more accurately as these instances are expected to be located "centrally" in the dataset, i.e., they appear in relatively dense regions of the data, see e.g. [25]. Therefore, we propose to use the following hubness-aware certainty score:

$$hc(x^*) = \frac{\big(N'_k(x^*)\big)^{\alpha} P(C') \prod\limits_{x_i \in \mathcal{N}_k(x^*)} P(x_i \in \mathcal{N}_k | C')}{\sum\limits_{C_j \in \mathcal{C}} \Big(P(C_j) \prod\limits_{x_i \in \mathcal{N}_k(x^*)} P(x_i \in \mathcal{N}_k | C_j) \Big)}, \tag{6}$$

where $N'_k(x^*)$ denotes how many times instance x^* appears as one of the k-nearest neighbors of other instances when considering the labeled training data $\mathcal{D}^{lab}$ together with the unlabeled instance x^*, i.e., $\mathcal{D}^{lab} \cup \{x^*\}$ and α is a hyper-parameter that controls the contribution of $N'_k(x^*)$ to the value of certainty score. Please note that in order to calculate $hc(x^*)$, we do not take other

Notation (cont.)

$N'_k(x)$ denotes how many times x occurs as one of the k-nearest neighbors of other instances when considering $\mathcal{D}^{lab} \cup \{x\}$.

unlabeled instances into account.

According to our empirical results (see Section 3), the above certainty estimation works well with $\alpha = 0.2$ in various domains ranging from breast cancer over colon cancer to lung cancer, therefore we use $\alpha = 0.2$ by default. In the example shown in Fig. 1, the above certainty estimate gives

$$\frac{2^{0.2} \times 0.2}{0 + 0.2} \approx 1.149$$

when classifying x_{11}, as x_{11} appears as nearest neighbor of x_6 and x_9 when considering all the eleven instances for the computation of the nearest neighbor relationships (we assume that the distance between x_{11} and x_9 is lower than the distance between x_9 and x_6, therefore, x_{11} will be the nearest neighbor of x_9 when considering all the instances).

2.4 Datasets and Methods for Evaluation

Datasets. We used publicly available gene expression data of breast cancer tissues [18], colon cancer tissues [1], and lung cancer tissues [2]. In these datasets, the expression levels of 7650, 6500 and 12,600 genes have been measured for 95, 62 and 203 patients respectively. The breast and colon cancer datasets had two classes, while the lung cancer dataset had five classes. In all the cases, classes correspond to subtypes of the disease or healthy tissues, see [9] for details. Out of the five classes of the lung cancer dataset, we ignored one because extraordinarily few instances (in particular, only six instances) belonged to that class.

Experimental protocol. We simulated two scenarios in which the available training data is not fully representative. In both scenarios, we selected a few instances as *labeled* training data while the remaining instances were considered as *unlabeled* data. The classifiers were evaluated on this unlabeled data. The

true class labels of the "unlabeled instances" were given in the datasets, however, these true class labels were only used for evaluation, i.e., the labels of the "unlabeled instances" were unknown to the classifier.

In the first scenario, denoted as BreastCancer-B, ColonCancer-B and LungCancer-B we considered five randomly selected instances per class as labeled training data. This results in *balanced* distribution of classes in the labeled training data whereas the entire datasets were class-imbalanced [9].

In the second scenario, denoted as BreastCancer-I, ColonCancer-I and LungCancer-I, we considered an *imbalanced* sample as labeled training data. In order to ensure a challenging classification task in which the labeled training data is not representative, we selected 5 instances from the majority class and 10 instances from the minority class(es) as labeled training data. By default, we report results observed in the first (balanced) scenario, unless the opposite is stated explicitly.

We repeated all the experiments 100 times with 100 different initial random selections of the labeled training instances. We measured the performance of the classifiers in terms of classification accuracy, i.e., the fraction of correctly classified unlabeled instances, macro-averaged F1-score and Matthews correlation coefficient (MCC). Both F1-score and MCC were aggregated over the runs and classes. We report the average and standard deviation of the accuracies achieved in the aforementioned 100 runs. Additionally, we used binomial test as suggested in [17], in order to judge if the differences between our approach and the baselines are statistically significant. We performed the aforementioned binomial test in each of the 100 runs and considered the difference to be statistically significant if the median of the resulting p-values was less than 0.05.

Compared Methods. We focus on the comparison of the following approaches:

- NHBNN-HS, i.e., NHBNN in self-training mode with the proposed hubness-

aware certainty score according to Formula (6),

- NHBNN-Simple, i.e., NHBNN in self-training mode with the straightforward certainty score according to Formula (5),
- k-NN in self-training mode with the proposed hubness-aware certainty score according to Formula (6),
- NHBNN-SV, i.e., supervised NHBNN that uses only the labeled training instances but does not learn from the unlabeled data,
- HFNN, i.e., Hubness-aware Fuzzy Nearest Neighbors, which is a hubness-aware supervised classifier, therefore, it uses only the labeled training instances but does not learn from the unlabeled data, see [26] for more details,
- GRF, i.e., semi-supervised classification with Gaussian Random Fields[1] based on [29].

Additionally, we run experiments with other classifiers, in particular SVMs and supervised k-NN.

In accordance with [22], by default, we used $k = 5$ for all the aforementioned variants of NHBNN and k-NN. Note, however, that we performed experiments with other k values as well and we observed similar trends. As distance measure, we used the Cosine distance with all the aforementioned classifiers.

For semi-supervised classifiers, by default, we report results for 20 iterations of self-training, i.e., 20 instances were labeled and added to the training set iteratively (one instance was labeled in each iteration) and then the model resulting after the 20th iteration was used to label all the remaining unlabeled instances.

[1] We predicted class labels according to Formula (5) in [29]. We note that in order to avoid numerical problems, we set GRF's length scale hyperparameters σ_d as 100-times the standard deviation of the d-th "component", which is the expression level of the d-th gene, in our case. In case of the binary classification tasks, we used the "default" decision threshold of 0.5. In case of the non-binary classification tasks, LungCancer-B and LungCancer-I, we used the one-vs-rest protocol with GRF.

3 Results

Table 1 and Table 2 show the accuracy and F1-score of our approach and the baselines. Our approach, NHBNN-HS, consistently outperforms all the examined baselines on all the three datasets in both scenarios. The only exception is in case of BreastCancer-I when NHBNN-HS performs slightly worse than NHBNN-Simple, although the difference is not significant statistically. We note that even in this case, NHBNN-HS significantly outperforms k-NN, HFNN and GRF. We observed similar trends when we evaluated our approach and the baselines in terms of MCC. Fig. 2 shows that NHBNN-HS systematically outperforms its competitors for various k values, except for $k = 1$. The diagrams in the top of Fig. 3 show the accuracy of our approach as function of α, i.e., the exponent of $N'_k(x^*)$ in Formula (6). As one can see, $\alpha = 0.2$ can be considered as a reasonable "default" setting of α. The diagrams in the bottom of Fig. 3 show the accuracy of our approach, NHBNN-HS, and NHBNN-Simple as function of the number of self-training iterations. For comparison, the accuracy of the NHBNN-SV is shown as well. As one can see, NHBNN-HS systematically outperforms NHBNN-Simple for various settings of the number of iterations.

Additionally, we tried (a) supervised k-NN and (b) support vector machines from the Weka software package [28] with polynomial and RBF kernels with various settings of the complexity constant and the exponent of the polynomial kernel. According to our observations, self-training was not able to substantially improve the performance of SVMs overall: SVMs without self-training performed as well as (or sometimes even better than) SVMs with self-training. More importantly, NHBNN-HS was competitive to SVMs, too: for example on the Breast Cancer and Colon Cancer datasets, best performing SVMs achieved classification accuracy of 0.781 and 0.705 respectively.

Despite the fact that cancer is a multifactorial disease, and therefore it is

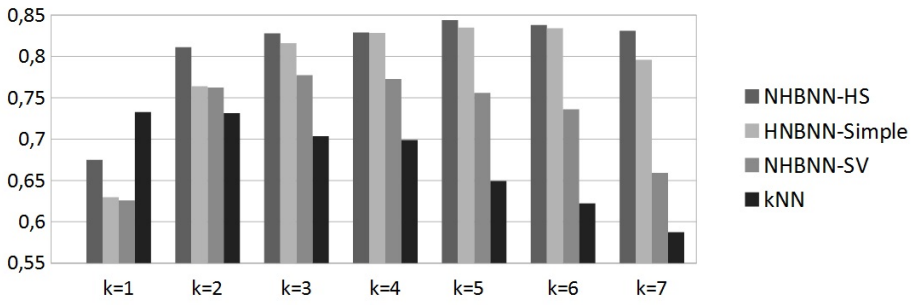

Figure 2: Accuracy of our approach, NHBNN-HS, and its competitors for various k values on the BreastCancer dataset.

inherently difficult, if not impossible, to determine the reason why an individual patient got the disease, we argue that the model built by NHBNN, i.e., the conditional probabilities describing how often characteristic patients (hubs) appear as nearest neighbors of patients from different classes, may be more interpretable to human experts than the model built by SVMs. Regarding supervised k-NN, we note that NHBNN-HS outperformed supervised k-NN as well which is in accordance with the previous results.

4 Discussion

As one can see from Table 1, both the algorithm and the certainty score are relevant: both NHBNN in self-training mode with the straightforward certainty score and k-NN with the hubness-aware certainty score achieve suboptimal accuracy compared with our approach NHBNN-HS. Furthermore, as we expected, semi-supervised classification outperforms supervised classification as it can be seen from the comparison against NHBNN-SV. These observations are confirmed by the results in case of various k values as shown in Fig. 2.

As one can see in the bottom of Fig. 3, on the Breast Cancer and Colon Cancer datasets NHBNN-HS and NHBNN-Simple converge to similar accura-

Table 1: Accuracy ± its standard deviation for our approach, NHBNN-HS, and the baselines averaged over 100 runs. Bold font denotes the best approach for each dataset. The symbols •/∘ denote if the differences between NHBNN-HS and its competitors are statistically significant (•) or not (∘).

	BreastCancer-B	ColonCancer-B	LungCancer-B
NHBNN-HS	**0.844 ± 0.040**	**0.808 ± 0.086**	**0.798 ± 0.128**
NHBNN-Simple	0.835 ± 0.049 ∘	0.790 ± 0.082 ∘	0.679 ± 0.114 •
k-NN	0.649 ± 0.155 •	0.650 ± 0.162 ∘	0.674 ± 0.329 ∘
NHBNN-SV	0.756 ± 0.103 ∘	0.637 ± 0.139 •	0.617 ± 0.125 •
HFNN	0.753 ± 0.107 ∘	0.633 ± 0.139 •	0.558 ± 0.130 •
GRF	0.619 ± 0.138 •	0.442 ± 0.154 •	0.621 ± 0.234 •
	BreastCancer-I	ColonCancer-I	LungCancer-I
NHBNN-HS	0.831 ± 0.080	**0.845 ± 0.035**	**0.876 ± 0.066**
NHBNN-Simple	**0.835 ± 0.065** ∘	0.817 ± 0.047 ∘	0.755 ± 0.086 •
k-NN	0.465 ± 0.251 •	0.615 ± 0.281 ∘	0.482 ± 0.335 •
NHBNN-SV	0.795 ± 0.093 ∘	0.719 ± 0.110 ∘	0.657 ± 0.103 •
HFNN	0.569 ± 0.185 •	0.477 ± 0.152 •	0.499 ± 0.125 •
GRF	0.275 ± 0.000 •	0.255 ± 0.000 •	0.094 ± 0.027 •

cies. In contrast, the proposed approach, NHBNN-HS converges to a much better solution on the Lung Cancer dataset.

Based on the observations above, we note that even in cases in which NHBNN-HS and NHBNN-Simple converge to the same solution, NHBNN-HS is preferable to NHBNN-Simple as (i) the former may lead to more accurate results if the number of self-training iterations is fixed or (ii) the same accuracy may be achieved in fewer self-training iterations. For example, on the Breast Cancer dataset, NHBNN-HS achieves an accuracy of 0.84 in just 13 iterations, whereas NHBNN-Simple requires 31 iterations to achieve the same accuracy, while on the Lung Cancer dataset, NHBNN-HS achieves an accuracy of 0.75 in 13 iterations, whereas NHBNN-Simple requires 36 iterations to achieve the same accuracy.

As shown in the top of Fig. 2, hyper-parameter α that controls the contribution of $N'_k(x^*)$ to the value of the certainty score effects the performance of

Table 2: Macro-averaged F1-scores of our approach, NHBNN-HS, and the baselines. Bold font denotes the best approach for each dataset.

	BreastCancer-B	ColonCancer-B	LungCancer-B
NHBNN-HS	**0.828**	**0.789**	**0.801**
NHBNN-Simple	0.817	0.781	0.756
k-NN	0.594	0.581	0.793
NHBNN-SV	0.746	0.642	0.729
HFNN	0.745	0.638	0.706
GRF	0.416	0.367	0.407
	BreastCancer-I	ColonCancer-I	LungCancer-I
NHBNN-HS	0.810	**0.806**	**0.823**
NHBNN-Simple	**0.814**	0.792	0.762
k-NN	0.521	0.645	0.727
NHBNN-SV	0.784	0.725	0.726
HFNN	0.669	0.619	0.686
GRF	0.216	0.203	0.399

the proposed approach which is in accordance with our expectations: setting $\alpha = 0$, the certainty scores of Formula (6) reduces to the straightforward certainty score of Formula (5). On the other hand, higher values of α result in increased influence of the $N'_k(x^*)$. While it is important to take $N'_k(x^*)$ into account in the certainty score, as our observations show, the balance between the hubness-score $N'_k(x^*)$ and the straightforward certainty scores leads to the overall best results.

Assuming that the distances between instances can be pre-calculated and cached, NHBNN-HS can be implemented with minimal additional computational costs compared with NHBNN-Simple. For each labeled instance, we only need to record the distance to its k-th nearest neighbor among the labeled instances. Let us call this distance the k-distance of a labeled instance. Let us consider a labeled instance x and an unlabeled instance x^*. By comparing the k-distance of x and the distance between x and x^*, one can simply decide if x^*

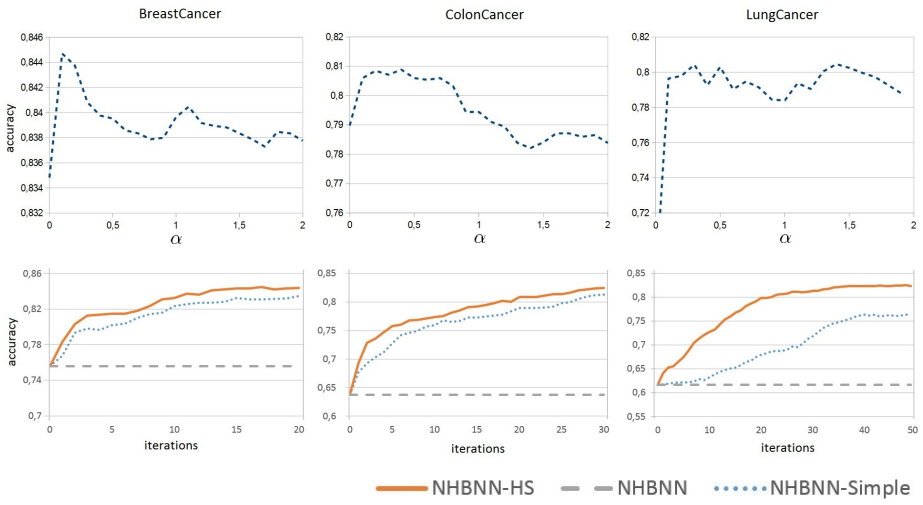

Figure 3: Accuracy of our approach, NHBNN-HS as function of: (i) α, i.e., the parameter controlling the contribution of $N'_k(x^*)$ to the certainty score (in the top), and (ii) the number of self-training iterations (in the bottom). Additionally, the accuracy of NHBNN-Simple and NHBNN-SV is shown in the diagrams in the bottom.

appears as one of the nearest neighbors of x when considering $\mathcal{D}^{lab} \cup \{x^*\}$. This way, $N'_k(x^*)$ can be calculated quickly. At the end of each self-training iteration, k-distances are to be updated based on the instance(s) that became labeled in that iteration. As these operations require minimal additional computational costs compared to other costs of the learning algorithm (such as distance calculations), for the same number of self-training iterations, the computational costs of NHBNN-HS and NHBNN-Simple are approximately the same. Taking the previous observations into account, we conclude that NHBNN-HS may achieve more accurate results with (approximately) the same computational costs, or the same accuracy with remarkably less computational costs.

While instances may influence classification decisions in many ways, hubs are generally known to play a crucial role in classification decisions. Specifically, in case of NHBNN, hubs influence the neighbour occurrence profiles of many

instances, i.e., they affect the conditional probabilities $P(x_i \in \mathcal{N}_k|C)$ of many instances.

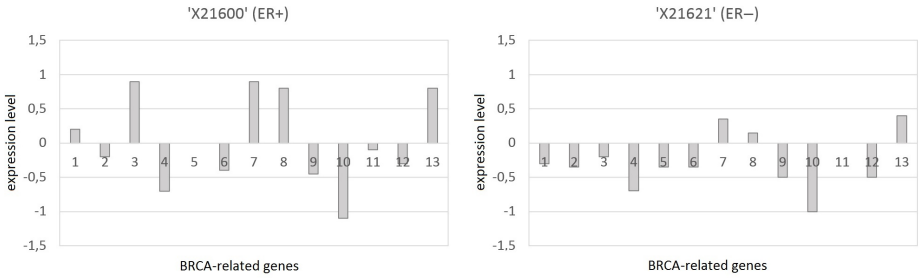

Figure 4: Excerpt from the gene expression profiles of two characteristic patients (hubs) of the Breast Cancer dataset.

To demonstrate that the proposed approach is indeed able to label hubs correctly, we selected two patients from the BreastCancer dataset, identified by X21600 and X21621 respectively. X21600 has ER+ subtype of breast cancer and appears as one of the k-nearest neighbors ($k = 5$) of 24 other ER+ patients, while it appears as one of the nearest neighbors of only *one* ER− patient. X21621 has ER− subtype of breast cancer and appears as one of nearest neighbors of 11 other patients, each of them having ER− subtype of breast cancer. The expression levels of the genes with descriptions containing "BRCA" is depicted in Figure 4 for these two patients. We considered the runs when these instances were not among the initially labeled instances and we observed that NHBNN-HS labeled X21621 always correctly, while it labeled X21600 in 97% of the aforementioned runs correctly. This illustrates that NHBNN-HS performs well in terms of labeling of the "most important" instances.

Next, we discuss the performance of GRF. One of the most important hyperparameters of GRF, which may affect its performance, is the decision threshold. In our experiments, we used the "default" value of 0.5, which is called *harmonic threshold* in [29]. This selection is in accordance with our assumption that only

a small set of labeled instances is given and this set is not a fully representative sample of the unlabeled data. On the other hand, in several practical applications, additional information might be available which allows to set GRF's decision threshold in a more informed way.

4.1 Concluding Remarks

In many applications, obtaining reliable class labels for large training samples may be difficult or even impossible. Therefore, semi-supervised classification techniques are required as they are able to take advantage of unlabeled data. Some of the most prominent recent methods developed for the classification of high-dimensional data follow the paradigm of hubness-aware data mining. However, hubness-aware classifiers have not been used for semi-supervised classification tasks previously. Therefore, in this paper, we introduced a semi-supervised hubness-aware classifier and we showed that it outperforms all the examined relevant baselines on the classification of gene expression data.

Based on the promising results presented in the paper, we envision that hubness-aware techniques will be used in further biomedical recognition tasks such as ECG-based person identification [7], diagnosis of schizophrenia [4] or link prediction in biomedical networks [27]. In order to accelerate this process, we made an implementation of hubness-aware machine learning techniques publicly available in the PyHubs software package on our website.[2] The PyHubs software package is implemented in Python, one of the most popular programming languages of data science. PyHubs may be seen as complementary to HubMiner [24] which is a Java-based implementation of hubness-aware machine learning techniques.

[2] http://www.biointelligence.hu/pyhubs

Acknowledgment

The Author thanks the anonymous Reviewers for their insightful comments. The PyHubs software package was implemented by Mararu-Nicoară Vlad under the supervision of the author of the current study. This research was performed within the framework of the grant of the Hungarian Scientific Research Fund - OTKA 111710 PD. This paper was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences.

References

[1] Uri Alon, Naama Barkai, Daniel A Notterman, Kurt Gish, Suzanne Ybarra, Daniel Mack, and Arnold J Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, 96(12):6745–6750, 1999.

[2] Arindam Bhattacharjee, William G Richards, Jane Staunton, Cheng Li, Stefano Monti, Priya Vasa, Christine Ladd, Javad Beheshti, Raphael Bueno, Michael Gillette, et al. Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses. *Proceedings of the National Academy of Sciences*, 98(24):13790–13795, 2001.

[3] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[4] Reza Boostani, Khadijeh Sadatnezhad, and Malihe Sabeti. An efficient classifier to diagnose of schizophrenia based on the eeg signals. *Expert Systems with Applications*, 36(3, Part 2):6492 – 6499, 2009.

[5] K. Buza, A. Nanopoulos, and L. Schmidt-Thieme. INSIGHT: Efficient and Effective Instance Selection for Time-Series Classification. In *15th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, volume 6635 of *Lecture Notes in Computer Science/Lecture Notes in Artificial Intelligence (LNCS/LNAI)*, pages 149–160. Springer, 2011.

[6] Olivier Chapelle, Bernhard Schölkopf, Alexander Zien, et al. *Semi-supervised learning*. MIT press Cambridge, 2006.

[7] Francesco Gargiulo, Antonio Fratini, Mario Sansone, and Carlo Sansone. Subject identification via ecg fiducial-based systems: influence of the type of qt interval correction. *Computer methods and programs in biomedicine*, 2015.

[8] Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid. Multimodal semi-supervised learning for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 902–909. IEEE, 2010.

[9] Wei-Jiun Lin and James J Chen. Class-imbalanced classifiers for high-dimensional data. *Briefings in bioinformatics*, 14(1):13–26, 2013.

[10] K. Marussy. The curse of intrinsic dimensionality in genome expression classification. In *Students' Scientific Conference, Budapest University of Technology and Economics*, 2014.

[11] Kristóf Marussy and Krisztian Buza. Success: A new approach for semi-supervised classification of time-series. In Leszek Rutkowski, Marcin Korytkowski, Rafa Scherer, Ryszard Tadeusiewicz, LotfiA. Zadeh, and JacekM. Zurada, editors, *Artificial Intelligence and Soft Computing*, volume 7894 of *Lecture Notes in Computer Science*, pages 437–447. Springer Berlin Heidelberg, 2013.

[12] M. Radovanović, A. Nanopoulos, and M. Ivanović. Nearest Neighbors in High-Dimensional Data: The Emergence and Influence of Hubs. In *Proceedings of the 26rd International Conference on Machine Learning (ICML)*, pages 865–872. ACM, 2009.

[13] M. Radovanović, A. Nanopoulos, and M. Ivanović. Hubs in Space: Popular Nearest Neighbors in High-Dimensional Data. *The Journal of Machine Learning Research (JMLR)*, 11:2487–2531, 2010.

[14] M. Radovanović, A. Nanopoulos, and M. Ivanović. Time-Series Classification in Many Intrinsic Dimensions. In *Proceedings of the 10th SIAM International Conference on Data Mining (SDM)*, pages 677–688, 2010.

[15] Miloš Radovanović. *Representations and Metrics in High-Dimensional Data Mining*. Izdavačka knjižarnica Zorana Stojanovića, Novi Sad, Serbia, 2011.

[16] I. Rish. An empirical study of the naive Bayes classifier. In *Proc. IJCAI Workshop on Empirical Methods in Artificial Intelligence*, 2001.

[17] Steven L Salzberg. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data mining and knowledge discovery*, 1(3):317–328, 1997.

[18] Christos Sotiriou, Soek-Ying Neo, Lisa M McShane, Edward L Korn, Philip M Long, Amir Jazaeri, Philippe Martiat, Steve B Fox, Adrian L Harris, and Edison T Liu. Breast cancer classification and prognosis based on gene expression profiles from a population-based study. *Proceedings of the National Academy of Sciences*, 100(18):10393–10398, 2003.

[19] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining*. Addison Wesley, 2005.

[20] Nenad Tomašev and Krisztian Buza. Hubness-aware knn classification of high-dimensional data in presence of label noise. *Neurocomputing*, 2015.

[21] Nenad Tomašev, Krisztian Buza, Kristóf Marussy, and Piroska B Kis. Hubness-aware classification, instance selection and feature construction: Survey and extensions to time-series. In *Feature Selection for Data and Pattern Recognition*, pages 231–262. Springer, 2015.

[22] N. Tomašev and D. Mladenić. Nearest neighbor voting in high dimensional data: Learning from past occurrences. *Computer Science and Information Systems*, 9:691–712, 2012.

[23] N. Tomašev, M. Radovanović, D. Mladenić, and M. Ivanović. A probabilistic approach to nearest neighbor classification: Naive hubness Bayesian k-nearest neighbor. In *Proceeding of the CIKM conference*, 2011.

[24] Nenad Tomašev. Hub miner v1.1, January 2015.

[25] Nenad Tomašev, Miloš Radovanović, Dunja Mladenić, and Mirjana Ivanovic. The role of hubness in clustering high-dimensional data. In *PAKDD (1)'11*, pages 183–195, 2011.

[26] Nenad Tomašev, Miloš Radovanović, Dunja Mladenić, and Mirjana Ivanović. Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. *International Journal of Machine Learning and Cybernetics*, 2013.

[27] Yuhao Wang and Jianyang Zeng. Predicting drug-target interactions using restricted boltzmann machines. *Bioinformatics*, 29(13):i126–i134, 2013.

[28] I.H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2011.

[29] Xiaojin Zhu, Zoubin Ghahramani, John Lafferty, et al. Semi-supervised learning using gaussian fields and harmonic functions. In *Proc. of the International Conference on Machine Learning*, volume 3, pages 912–919, 2003.