Agglomerative Multivariate Information Bottleneck

Noam Slonim Nir Friedman Naftali Tishby
School of Computer Science & Engineering, Hebrew University, Jerusalem 91904, Israel
{noamm, nir, tishby }@cs.huji.ac.il

Abstract

The *Information bottleneck method* is an unsupervised non-parametric data organization technique. Given a joint distribution $P(A, B)$, this method constructs a new variable T that extracts partitions, or clusters, over the values of A that are informative about B. In a recent paper, we introduced a general principled framework for multivariate extensions of the information bottleneck method that allows us to consider multiple systems of data partitions that are inter-related. In this paper, we present a new family of simple agglomerative algorithms to construct such systems of inter-related clusters. We analyze the behavior of these algorithms and apply them to several real-life datasets.

1 Introduction

The *Information bottleneck method* of Tishby et al [16] is an unsupervised non-parametric data organization technique. Given a joint distribution $P(A, B)$, this method constructs a new variable T that represents partitions of A which are informative about B. The construction of T attempts to find a tradeoff between the information we try to minimize, $I(T; A)$, and the information we try to maximize $I(T; B)$. This approach is useful in co-occurrence data, such as words and documents [14] where we want to capture what information one variable (e.g., use of a word) contains about the other (e.g., the document).

In a recent paper, Friedman et al. [5] introduce *multivariate* extension of the *IB* principle. This extension allows us to consider cases where the data partition is relevant with respect to several variables, or where we construct several systems of clusters simultaneously. In this framework, we specify the desired interactions by a pair of Bayesian networks. One network, G_{in}, represents which variables are compressed versions of the observed variables – each new variable compresses its parents in the network. The second network, G_{out}, represents which relations should be maintained or predicted.

Friedman et al. formulate the general principle as a tradeoff between the (multi) information each network carries. On the one hand, we want to minimize the information maintained by G_{in} and on the other to maximize the information maintained by G_{out}. They also provide a characterization of stationary points in this tradeoff as a set of *self-consistent* equations. Moreover, they prove that iterations of these equations converges to a (local) optimum. Then, they describe a *deterministic annealing* procedure that constructs a solution by tracking the bifurcation of clusters as it traverses the tradeoff curve, similar to the original *IB* method.

In this paper, we consider alternative approaches to solving multivariate *IB* problems.

This is motivated by the success of the *agglomerative IB* of Slonim and Tishby [13]. As they show, a bottom-up greedy agglomeration is a simple heuristic procedure that can yield good solutions to the original *IB* problem. Here we extend this idea in the context of multivariate *IB* problems. We start by analyzing the cost of agglomeration steps within this framework. This both elucidates the criteria that guides greedy agglomeration, and provides for efficient *local* evaluation rules for agglomeration steps. This construction results with a novel family of information theoretic agglomerative clustering algorithms, that can be specified using the graphs G_{in} and G_{out}. We demonstrate the performance of some of these algorithms for document and word clustering and gene expression analysis.

2 Multivariate Information Bottleneck

A *Bayesian network structure* G is a DAG that specifies interactions among variables [9]. A distribution P is *consistent* with G (denoted $P \models G$), if $P(X_1, \ldots, X_n) = \prod_i P(X_i \mid \mathbf{Pa}^G_{X_i})$, where $\mathbf{Pa}^G_{X_i}$ are the parents of X_i in G. Our main interest is in the information that the variables $X_1, \ldots, X_n$ contain about each other. A quantity that captures this is the *multi-information* given by $\mathcal{I}(X_1, \ldots, X_n) = \mathcal{D}(P(X_1, \ldots, X_n) \| P(X_1) \cdots P(X_n))$, where $\mathcal{D}(p\|q)$ is the familiar Kullback-Liebler divergence [2].

Proposition 2.1 [5] *Let G be a DAG over $\{X_1, \ldots, X_n\}$, and let $P \models G$ be a distribution. Then, $\mathcal{I}^G \equiv \mathcal{I}(X_1, \ldots, X_n) = \sum_i I(X_i; \mathbf{Pa}^G_{X_i})$.*

That is, the multi-information is the sum of *local* mutual information terms between each variable and its parents (denoted $\mathcal{I}^G$).

Friedman et al. define the multivariate *IB* problem as follows. Suppose we are given a set of observed variables, $\mathbf{X} = \{X_1, \ldots, X_n\}$ and their joint distribution $P(X_1, \ldots, X_n)$. We want to "construct" new variables $\mathbf{T}$, where the relations between the observed variables and these new compression variables are specified using a DAG G_{in} over $\mathbf{X} \cup \mathbf{T}$ where the variables in $\mathbf{T}$ are leafs. Thus, each T_j is a stochastic function of a set of variables $\mathbf{U}_j = \mathbf{Pa}^{G_{in}}_{T_j} \subseteq \mathbf{X}$. Once these are set, we have a joint distribution over the combined set of variables: $P(\mathbf{X}, \mathbf{T}) = P(\mathbf{X}) \prod_j P(T_j \mid \mathbf{U}_j)$.

The "relevant" information that we want to preserve is specified by another DAG, G_{out}. This graph specifies, for each T_j which variables it predicts. These are simply its children in G_{out}. More precisely, we want to predict each X_i (or T_j) by $\mathbf{V}_{X_i} = \mathbf{Pa}^{G_{out}}_{X_i}$ (resp. $\mathbf{V}_{T_j} = \mathbf{Pa}^{G_{out}}_{T_j}$), its parents in G_{out}. Thus, we think of $\mathcal{I}^{G_{out}}$ as a measure of how much information the variables in $\mathbf{T}$ maintain about their target variables.

The Lagrangian can then be defined as

$$\mathcal{L}[p(T_1 \mid \mathbf{U}_1), ..., p(T_k \mid \mathbf{U}_k)] = \mathcal{I}^{G_{out}} - \mathcal{T} \cdot \mathcal{I}^{G_{in}} , \tag{1}$$

with a *tradeoff parameter (Lagrange multiplier)* $\mathcal{T}$. The variation is done subject to the normalization constraints on the partition distributions. Thus, we balance between the information $\mathbf{T}$ loses about $\mathbf{X}$ in G_{in} and the information it preserves in G_{out}.

Friedman et al. [5] show that stationary points of this Lagrangian satisfy a set of self-consistent equations. Moreover, they show that iterating these equations converges to a stationary point of the tradeoff. Then, extending the procedure of Tishby et al [16], they propose a procedure that searches for a solution of the *IB* equations using a 'deterministic annealing' approach [11]. This is a top-down hierarchical algorithm that starts from a single cluster for each T_j at $\mathcal{T} \to \infty$, and then undergoes a cascade of cluster splits as $\mathcal{T}$ is being "cooled". These determines "soft" trees of clusters (one for each T_j) that describe solutions at different tradeoff values of $\mathcal{T}$.

3 The Agglomerative Procedure

For the original *IB* problem, Slonim and Tishby [13] introduced a simpler procedure that performs greedy bottom-up merging of values. Several successful applications of this algorithm are already presented for a variety of real-world problems [12, 14, 15, 17]. The main focus of the current work is in extending this approach for the multivariate *IB* problem. As we will show, this will lead to further insights about the method, and also provide a rather simple and intuitive clustering procedures.

We consider procedures that start with a set of clusters for each T_j (usually the most fine-grained solution we can consider where $T_j = \mathbf{U}_j$) and then iteratively reduce the cardinality of one of the T_j's by *merging* two values t_j^l and t_j^r of T_j into a single value $\bar{t}_j$. To formalize this notion we must define the membership probability of a new cluster $\bar{t}_j$, resulting from merging $\{t_j^l, t_j^r\} \Rightarrow \bar{t}_j$ in T_j. This is done rather naturally by

$$p(\bar{t}_j \mid \mathbf{U}_j) = p(t_j^l \mid \mathbf{U}_j) + p(t_j^r \mid \mathbf{U}_j) \quad . \tag{2}$$

In other words, we view the event $\bar{t}_j$ as the union of the events t_j^l and t_j^r.

The crucial question is of course which pair to merge at each step. We know that the merger "cost" in our terms is exactly the difference in the values of $\mathcal{L}$, before and after the merger. Let T_j^{bef} and T_j^{aft} denote the random variables that correspond to T_j, before and after the merger, respectively. Thus, the values of $\mathcal{L}$ before and after the merger are calculated based on T_j^{bef} and T_j^{aft}. The merger cost is then simply given by,

$$\Delta\mathcal{L}(t_j^l, t_j^r) = \mathcal{L}^{bef} - \mathcal{L}^{aft} \quad . \tag{3}$$

The greedy procedure evaluates all the potential mergers (for all T_j) and then applies the best one (i.e., the one that minimizes $\Delta\mathcal{L}(t_j^l, t_j^r)$). This is repeated until all the variables in $\mathbf{T}$ degenerate into trivial clusters. The resulting set of trees describes a range of solutions at different resolutions.

This agglomerative approach is different in several important aspects from the deterministic annealing approach described above. In that approach, by cooling $\mathcal{T}$, we move along a tradeoff curve from the trivial - single cluster - solution toward solutions with higher resolutions that preserve more information in G_{out}. In contrast, in the agglomerative approach we progress in the opposite direction. We start with a high resolution clustering and as the merging process continues we move toward more and more compact solutions. During this process $\mathcal{T}$ is kept constant and the driving force is the reduction in the cardinality of the T_j's. Additionally, the merging does not attempt directly to maintain the (stationary) self-consistent "soft" membership probabilities. Thus, we do not expect the self-consistent equations to hold at solutions found by the agglomerative procedure. On the other hand, the agglomerative process is much simpler to implement and fully deterministic. As we will show, it provides sufficiently good solutions for the *IB* problem in many situations.

4 Local Merging Criteria

In the procedure we outline above, at every step there are $O(|T_j|^2)$ possible mergers of values of T_j. A direct calculation of the costs of all these potential mergers is typically infeasible. However, it turns out that one may calculate $\Delta\mathcal{L}(t_j^l, t_j^r)$ while examining only the probability distributions that involve t_j^l and t_j^r directly. Generalizing the results of [13] for the original *IB*, we now develop a closed-form formula for $\Delta\mathcal{L}(t_j^l, t_j^r)$.

To describe this result we need the following definition. The *Jensen-Shannon (JS) divergence* [8, 4] between two probabilities $p_1(X), p_2(X)$ is given by $JS_\Pi(p_1, p_2) =$

$\pi_1 KL[p_1(X)\|\bar{p}(X)] + \pi_2 KL[p_2(X)\|\bar{p}(X)]$, where $\Pi = \{\pi_1, \pi_2\}$ is a normalized probability and $\bar{p}(X) = \pi_1 p_1(X) + \pi_2 p_2(X)$. The JS divergence is equal zero if and only if both its arguments are identical. It is upper bounded and symmetric, though it is not a metric. One interpretation of the JS-divergence is as the (logarithmic) measure of the likelihood that the two sample distributions originate by the most likely common source, denoted by $\bar{p}$. In addition, we need the notation $\mathbf{V}_{X_i}^{-j} = \mathbf{V}_{X_i} - \{T_j\}$ (similarly for $\mathbf{V}_{T_\ell}^{-j}$).

Theorem 4.1 *Let $t_j^l, t_j^r \in T_j$ be two clusters. Then, $\Delta\mathcal{L}(t_j^l, t_j^r) = p(\bar{t}_j) \cdot d(t_j^l, t_j^r)$ where*

$$\begin{aligned} d(t_j^l, t_j^r) \quad \equiv \quad & \sum_{i:T_j \in \mathbf{V}_{X_i}} E_{P(\cdot|\bar{t}_j)}[JS_{\Pi_{\mathbf{V}_{X_i}^{-j}}}(p(X_i \mid t_j^l, \mathbf{V}_{X_i}^{-j}), p(X_i \mid t_j^r, \mathbf{V}_{X_i}^{-j}))] \\ + \quad & \sum_{\ell:T_j \in \mathbf{V}_{T_\ell}} E_{P(\cdot|\bar{t}_j)}[JS_{\Pi_{\mathbf{V}_{T_\ell}^{-j}}}(p(T_\ell \mid t_j^l, \mathbf{V}_{T_\ell}^{-j}), p(T_\ell \mid t_j^r, \mathbf{V}_{T_\ell}^{-j}))] \\ + \quad & JS_\Pi(p(\mathbf{V}_{T_j} \mid t_j^l), p(\mathbf{V}_{T_j} \mid t_j^r)) - \mathcal{T} \cdot JS_\Pi(p(\mathbf{U}_j \mid t_j^l), p(\mathbf{U}_j \mid t_j^r)) \ . \end{aligned}$$

and $\Pi_{\mathbf{z}} = \{\pi_{l,\mathbf{z}},\ \pi_{r,\mathbf{z}}\} = \{p(t_j^l \mid \bar{t}_j, \mathbf{z}), p(t_j^r \mid \bar{t}_j, \mathbf{z})\}$.

A detailed proof of this theorem will be given elsewhere. Thus, the merger cost is a multiplication of the weight of the merger components ($p(\bar{t}_j)$) with their "distance" given by $d(t_j^l, t_j^r)$. Notice that due to the properties of the JS-divergence, this distance is symmetric. In addition, the last term in this distance, has the opposite sign to the first three terms. Thus, the distance between two clusters is a tradeoff between these two factors. Roughly speaking, we may say that the distance is minimized for pairs that give similar predictions about the variables connected with T_j in G_{out} and have different predictions (minimum overlap) about the variables connected with T_j in G_{in}. We notice also the analogy between this result and the main theorem in [5]. In [5] the optimization is governed by the KL divergences between data and clusters centroid, or by the likelihood that the data was generated by the centroid distribution. Here the optimization is controlled through the JS divergences, i.e. the likelihood that the two clusters have a common source.

Next, we notice that after applying a merger, only a small portion of the other mergers costs change. The following proposition characterizes these costs.

Proposition 4.2 *The merger $\{t_j^l, t_j^r\} \Rightarrow \bar{t}_j$ in T_j can change the cost $\Delta\mathcal{L}(t_\ell^l, t_\ell^r)$ only if $p(\bar{t}_j, \bar{t}_\ell) > 0$ and T_j, T_ℓ co-appear in some information term in $\mathcal{I}^{G_{out}}$.*

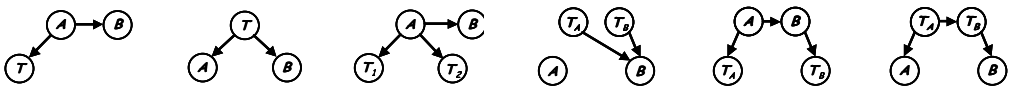
This proposition is particlarly useful, when we consider "hard" clustering where T_j is a function of $\mathbf{U}_j$. In this case, $p(\bar{t}_j, \bar{t}_\ell)$ is often zero (especially when T_j and T_ℓ compressing similar variables, i.e., $\mathbf{U}_j \cap \mathbf{U}_\ell \neq \emptyset$). In particular, after the merger $\{t_j^l, t_j^r\} \Rightarrow \bar{t}_j$, we do not have to reevaluate merger costs of other values of T_j, except for mergers of $\bar{t}_j$ with each of these values.

In the case of hard clustering we also find that $I(T_j; \mathbf{U}_j) = H(T_j)$ (where $H(p)$ is Shannon's entropy). Roughly speaking, we may say that $H(p)$ is decreasing for less balanced probability distributions p. Therefore, increasing $\mathcal{T}$ will result with a tendency to look for less balanced "hard" partitions and vice verse. This is reflected by the fact that the last term in $d(t_j^l, t_j^r)$ is then simplified through $JS_\Pi(p(\mathbf{U}_j \mid t_j^l), p(\mathbf{U}_j \mid t_j^r)) = H(\Pi)$.

5 Examples

We now briefly consider three examples of the general methodology. For brevity we focus on the simpler case of hard clustering. We first consider the example shown in figure 1(a). This choice of graphs results in the *original IB* problem. The merger cost in this case is given by,

$$\Delta\mathcal{L}(t^l, t^r) = p(\bar{t}) \cdot (JS_\Pi(p(B \mid t^l), p(B \mid t^r)) - \mathcal{T} H(\Pi)) \ . \tag{4}$$



G_{in} G_{out} G_{in} G_{out} G_{in} G_{out}

$I(T;B) - \mathcal{T} I(T;A)$

$I(T_1, T_2; B)$
$-\mathcal{T}(I(T_1;A) + I(T_2;A))$

$I(T_A; T_B)$
$-(\mathcal{T}-1)(I(T_A;A) + I(T_B;B))$

(a) Original Bottleneck (b) Parallel Bottleneck (c) Symmetric Bottleneck

Figure 1: The source and target networks and the corresponding Lagrangian for the three examples we consider.

Note that for $\mathcal{T} = 0$ we get exactly the algorithm presented in [13].

One simple extension of the original *IB* is the *parallel* bottleneck [5]. In this case we introduce two variables T_1 and T_2 as in Figure 1(b), both of them are functions of A. Similarly to the original *IB*, G_{out} specifies that T_1 and T_2 should predict B. We can think of this requirement as an attempt to decompose the information A contains about B into two 'orthogonal' components. In this case, the merger cost for T_1 is given by,

$$\Delta\mathcal{L}(t_1^l, t_1^r) = p(\bar{t}_1) \cdot (E_{P(\cdot|\bar{t}_1)}[JS_{\Pi_{T_2}}(p(B \mid t_1^l, T_2), p(B \mid t_1^r, T_2))] - \mathcal{T} H(\Pi)) \ . \quad (5)$$

Finally, consider the *symmetric* bottleneck [5, 14]. In this case, we want to compress A into T_A and B into T_B so that T_A extracts the information A contains about B, and at the same time T_B extracts the information B contains about A. The DAG G_{in} of figure 1(c) captures the form of the compression. The choice of G_{out} is less obvious and several alternatives are described in [5]. Here, we concentrate only in one option, shown in figure 1(c). In this case we attempt to make each of T_A and T_B sufficient to separate A from B. Thus, on one hand we attempt to compress, and on the other hand we attempt to make T_A and T_B as informative about each other as possible. The merger cost in T_A is given by

$$\Delta\mathcal{L}(t_A^l, t_A^r) = p(\bar{t}_A) \cdot JS_{\Pi}(p(T_B \mid t_A^l), p(T_B \mid t_A^r)) - (\mathcal{T}-1)H(\Pi)) \ , \quad (6)$$

while for merging in T_B we will get an analogous expression.

6 Applications

We examine a few applications of the examples presented above. As one data set we used a subset of the 20 newsgroups corpus [7] where we randomly choose 2000 documents evenly distributed among the 4 science discussion groups (*sci.crypt, sci.electronics, sci.med* and *sci.space*). [1] Our pre-processing included ignoring file headers, lowering upper case and ignoring words that contained non 'a..z' characters. Given this document set we can evaluate the joint probability $p(W, D)$, which is the probability that a random word position is equal to $w \in W$ and at the same time the document is $d \in D$. We sort all words by their contribution to $I(W; D)$ and used only the 2000 'most informative' ones, ending up with a joint probability with $|W| = |D| = 2000$.

We first used the original *IB* to cluster W, while trying to preserve the information about D. This was already done in [14] with $\mathcal{T} = 0$, but in this new experiment we took $\mathcal{T} = 0.15$. Recall that increasing $\mathcal{T}$ results in a tendency for finding less balanced clusters.

[1] We used exactly the same subset already used in [14].

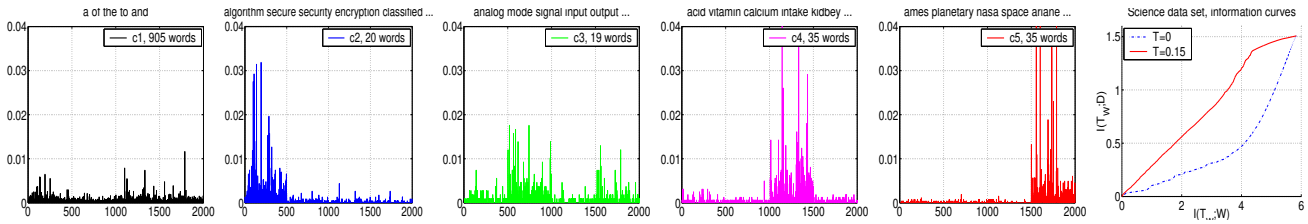


Figure 2: $P(D \mid T_W)$ for 5 word clusters in T_W. Documents $1-500$ belong to *sci.crypt* category, $501-1000$ to *sci.electronics* , $1001-1500$ to *sci.med* and $1501-2000$ to *sci.space*. In the title of each panel we see the 5 most frequent words in the cluster. The 'big' cluster (left panel) is clearly less informative about the structure of D. In the right panel we see the two information curves. Given some compression level, for $\mathcal{T} = 0.15$ we preserve much more information about D than for $\mathcal{T} = 0$.

Indeed, while for $\mathcal{T} = 0$ we got relatively balanced word clusters (high $H(T_W)$), for $\mathcal{T} = 0.15$ the probability $p(T_W)$ is much less smooth. For 50 word clusters, one cluster contained almost half of the words, while the other clusters were typically much smaller. Since the algorithm also tries to maximize $I(T_W; D)$, the words merged into the big cluster are usually the less informative words about D. Thus, a word must be highly informative to stay out of this cluster. In this sense, increasing $\mathcal{T}$ is equivalent for inducing a "noise filter", that leave only the most informative features in specific clusters. In figure 2 we present $p(D \mid t_W)$ for several clusters $t_W \in T_W$. Clearly, words who passed the "filter" form much more informative clusters about the real structure of D. A more formal demonstration of this effect is given in the right panel of figure 2. For a given compression level (i.e. a given $I(T_W; W)$), we see that taking $\mathcal{T} = 0.15$ preserve much more information about D.

While an 'exact' implementation of the symmetric *IB* will require alternating mergers in T_W and T_D, an approximated approach require only two steps. First we find T_W. Second, we project each $d \in D$ into the low dimensional space defined by T_W, and use this more robust representation to extract *document* clusters T_D. Approximately, we are trying to find T_W and T_D that will maximize $I(T_W; T_D)$. This two-phase *IB* algorithm was shown in [14] to be significantly superior to six other common document clustering methods, when the performance are measured by the correlation of the obtained document clusters with the real newsgroup categories. Here we use the same procedure, but for finding T_W we take $\mathcal{T} = 0.15$ (instead of zero). Using the above intuition we predict that $\mathcal{T} > 0$ can induce a cleaner representation for the document set. Indeed, the averaged correlation of T_D (for $|T_D| = 4$) with the original categories was 0.65, while for $\mathcal{T} = 0$ it was 0.58 (the average is taken over different number of word clusters, $|T_W| = 10, 11...50$). Similar results were obtained for all the 9 other subsets of the 20 newsgroups corpus described in [14].

Finally, to demonstrate the performance of the *parallel IB* we apply it to gene expression measurements of ~ 6800 genes in 72 samples of Leukemia [6]. We removed genes that were not expressed in the data and normalized the measurements of each sample to get a joint probability $P(G, A)$ over genes and samples (with uniform prior on samples). We sorted all genes by their contribution to $I(G; A)$ and chose the 500 most informative ones, which capture 47% of the original information, ending up with a joint probability with $|A| = 72$ and $|G| = 500$. Using the *parallel IB* algorithm (with $\mathcal{T} = 0$) we clustered the arrays A into two clustering hierarchies, T_1 and T_2, that try *together* to capture the information about G. For $|T_j| = 4$ we find that each $I(T_j; G)$ preserve about 15% of the original information. However, taking $|T_j| = 2$ (i.e. again, just 4 clusters) we see that the combination of the hierarchies, $I(T_1, T_2; G)$, preserve 21% of the original information. We then compared the partitions we found against sample annotations which included type of

leukemia (*ALL* vs. *AML*), type of cells, source of sample, and donating hospital. We found that the first split of the first hierarchy almost perfectly match the split between B-cells and T-cells (among the 47 samples for which we had this annotation). The first split of the second hierarchy separates a cluster of 18 samples, almost all of which are *ALL* samples taken from the bone marrow of patients from the same hospital. These results demonstrate the ability of the algorithm to extract simultaneously different meaningful independent partitions of the data. A more detailed analysis of these results will be presented elsewhere.

7 Discussion

The analysis presented by this work enables to implement a family of novel agglomerative clustering algorithms. All of these algorithms are leaning on the same optimization framework described by the *multivariate IB* method.

The suggested approach has several advantages over the 'deterministic annealing' algorithms suggested in [5]. It is simpler and fully deterministic. There is no need to identify cluster splits which is usually rather tricky. Though agglomeration procedures do not scale linearly with the sample size as top down methods do, there exist several heuristics to improve the complexity of these algorithms (e.g. [1]).

While a typical initialization of an agglomerative procedure induces "hard" clustering solutions, all of the above analysis holds for "soft" clustering as well. Moreover, as already noted in [13], the obtained "hard" partitions can be used as a platform to find also "soft" solutions through a process of "reverse annealing". This raises the possibility for using an agglomerative procedure over "soft" clustering solutions, which we we leave for future work.

We could describe here only a few relatively simple examples. These examples show promising results on non trivial real life data. Moreover, other choices of G_{in} and G_{out} can yield additional novel algorithms with applications over a variety of data types.

References

[1] L. D. Baker and A. K. McCallum. Distributional clustering of words for text classification. In *ACM SIGIR 98*. 1998.

[2] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. 1991.

[3] T. Hofmann. Probabilistic latent semantic indexing. In *ACM SIGIR 99*, pages 50–57. 1999.

[4] R. El-Yaniv, S. Fine, and N. Tishby. Agnostic classification of Markovian sequences. In NIPS'97.

[5] N. Friedman, O. Mosenzon, N. Slonim and N. Tishby Multivariate Information Bottleneck UAI, 2001, to appear. See `www.cs.huji.ac.il/~nir/publications.html`.

[6] T. Golub, D. Slonim, P. Tamayo, C.M. Huard, J.M. Caasenbeek, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring *Science* **286**, 531–537, 1999.

[7] K. Lang. Learning to filter netnews. In ICML'95.

[8] J. Lin. Divergence Measures Based on the Shannon Entropy. *IEEE Trans. Info. Theory*, 37(1):145–151, 1991.

[9] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. 1988.

[10] F.C. Pereira, N. Tishby, and L. Lee. Distributional clustering of english words. In ACL'93.

[11] K. Rose. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proc. IEEE*, 86:2210–2239, 1998.

[12] N. Slonim, R. Somerville, N. Tishby, and O. Lahav. Objective spectral classification of galaxies using the information bottleneck method. in "Monthly Notices of the Royal Astronomical Society", MNRAS, 323, 270, 2001.

[13] N. Slonim and N. Tishby. Agglomerative Information Bottleneck. In NIPS'99.

[14] N. Slonim and N. Tishby. Document clustering using word clusters via the information bottleneck method. In *ACM SIGIR 2000*, 2000.

[15] N. Slonim and N. Tishby. The power of word clusters for text classification. In *ECIR*, 2001.

[16] N. Tishby, F. Pereira, and W. Bialek. The Information Bottleneck method. In *Proc. 37th Allerton Conference on Communication and Computation*. 1999.

[17] N. Tishby and N. Slonim. Data clustering by markovian relaxation and the information bottleneck method. In NIPS'00.