Structure Based Computational Aspects of Similarity and Analogy in Natural Language

A thesis submitted for the degree of Doctor of Philosophy

By Zvika Marx

Submitted to the Senate of the Hebrew University in the year 2004

This work was carried out under the supervision of

Prof. Eli Shamir and Dr. Ido Dagan

Abstract

The ability to identify analogies and correspondences is one of the fascinating aspects of intelligence. It allows learning across different situations, systems and domains, where the common base to learning is not trivial or immediate. The research in cognitive science has acknowledged the significance of analogy making to human thinking. Several previous works on analogy making suggested computational mechanisms for constructing detailed mapping that connects corresponding ingredients across a given pair of analogized systems.

In this work, we introduce a new approach to understanding, identifying and forming analogies and correspondences. In distinction from previous works on analogies, our approach and the computational methods derived from it are applicable to real world problems, such as the task of identification of corresponding topics in texts on different domains. This work thus bridges between cognitive observations regarding analogy making, which inspired this work but provided no concrete utilizable computational recipes, with techniques that have been proven efficient in processing real world data.

The methods introduced in this work extend the well known data clustering problem. The key mechanism used to identify correspondences through clustering is directing corresponding data elements, from different subsets of elements each representing one of the systems between which the correspondence is being drawn, to be included in the same cluster. The straightforward application of a standard clustering technique would not address well this target: a standard clustering method would often produce clusters with elements of only one of the representative subsets, particularly when these subsets are relatively homogenous. Our methods, however, are specifically designed to cluster together corresponding elements from both subsets while neutralizing the impact of homogeneity within each subset.

The first method that we introduce, termed *coupled clustering*, addresses the problem of partitioning two representative element subsets. This method extends an axiomatic framework of similarity-based data clustering (Puzicha, Hofmann & Buhmann, 2000). The other method, *cross-partition clustering*, modifies and generalizes the coupled clustering setting along several aspects: it is based on vectorial representation of the given data rather than on pairwise proximity values, it produces soft (probabilistic) rather than deterministic partitioning and it allows revealing correspondences across more than two subsets. The cross-partition clustering method is based on the *information bottleneck*

(Tishby, Pereira & Bialek, 1999) and *information distortion* (Gedeon, Parker & Dimitrov, 2003) methods, which are grounded on information theoretic approach to data clustering.

The setting underlying our approach is considerably different than previous views of analogy making. The two methods that we introduce ascribe the correspondence being formed to a counterbalance between different factors. In coupled clustering, these factors are shared pairwise similarity (across subsets) and prominence of the formed cluster in each one of both analogized subsets. In the cross partition method, the underlying factors are communal feature distribution patterns versus the independence of these patterns on the pre-partition of the data to distinct subsets between which a correspondence is revealed.

Both methods were developed using general formulation. They are capable of identifiable correspondences drawn across any sets of data elements that are represented through feature vectors or a similarity matrix, regardless of the source of the data. Hence in principle, they are applicable to a large variety of problems and domains. In this work both methods were applied successfully to synthetic and textual data.

The textual experiments addressed the task of identifying corresponding sub-topics across related but distinct domains, each represented through a set of domain-related keywords extracted from appropriate corpora. The similarity measure and vectorial representations required as input to our framework were compiled based on word co-occurrence statistics. Most experiments were focused on identifying correspondence between different religions: Buddhism, Christianity, Hinduism, Islam and Judaism. With no prior specialization or training in the study of religions, our methods identified analogous factors shared by several religions in varying levels of resolution: “spiritual” versus “practical” dimensions in a coarse view and aspects such as “sacred writings”, “rite and festivals” and “sin and suffering” in more detailed level. These findings are in apparent agreement with comparative religion studies that were based on a comparable approach. For the purpose of systematic evaluation, we have measured the overlap between our outcome and religion-related term clusters provided by experts. The match between the experts’ clusters and the outcome of our method was very close to the level of agreement between the experts.

Acknowledgments

I would like to thank my thesis co-advisors, Ido Dagan and Eli Shamir. I enjoyed very much their supervision and I learned a lot from both of them although (and because) working with each one of them was a very different experience.

I would like to express my gratitude to several additional individuals who helped with advice and discussion:

Shaike Shen who introduced me to the domain of metaphor and analogy research.

Joachim Buhmann, who was kind to host and guide me during the earlier period of the work; Chapter 4 of this thesis is the result of this collaboration.

Chapter 4 gained also from the guidance of Jan Puzicha regarding the implementation of his approach to similarity.

Naftali Tishby with whom I had several interesting discussions along the various stage of the work.

Yuval Krymolowski who listened to my ideas, including ones that did not make much sense, with a lot of attention and collaborated with me throughout most of the time I worked on this thesis.

Ran Gilad-Bachrach who made valuable comments and suggestions regarding Chapter 5 of the thesis.

Tomáš Gedeon, who read and commented too on Chapter 5.

Ulrike Baldewein and Beata Beigman Klebanov who read the thesis draft and commented on it; many thanks to Beata also for her kind help in printing the version submitted for review.

William Shepard, Tiina Mahlamäki, Ilkka Pyysiainen, Rudi Siebert and Eitan Reich who responded, as experts in the comparative study of religions, to our survey on key-terms used in this field; with most of them I had further illuminating discussions regarding their domain of expertise.

Finally, I would like to thank and to express my love to the most significant individuals in my life: my parents Esther and Tommi Marx, my kids Dor, Ofri and Na’ama and my beloved and supporting wife Daphna.

Table of Contents

Chapter 1: Introduction

Identification of analogies and correspondences has consistently enlightened various fields of knowledge and scholarship. Historical situations and events, for instance, provide a rich field for the construction of analogies. In a unique enterprise, which dates back to Plutarch's "Parallel Lives", each member in a list of Greek public figures is paired with a Roman counterpart, whose position, actions and life events match in an illuminative manner.[1] This classical piece demonstrates one of the fascinating aspects of human intelligence: acquiring knowledge in a context of a particular object – be it a person, an event or a whole domain – allows applying this knowledge and enriching it in a related but different context. To this end, it is required to identify those relevant aspects that correspond to each other across the objects being studied. In other words, utilizing and enriching knowledge along different contexts is grounded on the ability to perceive some compound relation, an equivalence or analogy, between the contexts.

As the ability to perceive correspondence is fundamental to human intelligence, it is interesting to study computational mechanisms underlying it. Along with the theoretical merit, computational methods enabling the automated detection of analogies might turn highly practical in the current information overload era. Consider, for example, the following two text fragments extracted from a pair of 1986 Reuters' news-articles:

1. LOS ANGELES, March 13 – *Computer Memories Inc*. ... agreed to **acquire** *Hemdale Film Corp*. ... That company's **owner**, *John Daly*, would then ***become*** **chief executive officer** of the combined company...
2. NEW YORK, March 25 – *Messidor Ltd* said it signed a letter of intent to **acquire** 100 pct of the outstanding shares of *Triton Beleggineng Nederland B.V.* ... If approved, the **president** of *Triton*, *Hendrik Bokma*, will be ***nominated*** as **chairman** of the combined company. ...

The similarity between the above fragments is apparent: they both deal with the intention of a company to acquire another company. Typically, computational methods for assessing similarity between documents rely on the proportion of shared terms or keywords. In our example the word '*acquire*' appears in both articles, so keyword-based methods would count it as a positive evidence

[1]Plutarch's "Lives" can be browsed at **http://classics.mit.edu/Browse/browse-Plutarch.html.**

for evaluating the text fragments as similar to each other. More sophisticated methods (e.g. *Latent Semantic Indexing*; Deerwester et al., 1990) incorporate term-similarity models that may take into account correspondence of different terms that resemble in their meaning. Thus, corresponding terms such as '*owner*' — '*president*' and '*chief executive officer*' — '*chairman*' may contribute to the unified value of evaluated similarity.

Now, consider yet another pair of terms from the above fragments: '*become*' — '*nominated*'. These terms probably share only a moderate degree of similarity in general, but a human reader will find that they meaningfully correspond to one another in this particular context. Identification of this context-dependent equivalence also enables a reader to perceive that *John Daly* and *Hendrik Bokma* – names that the reader is likely not to encounter before – play in the above texts an analogous part of being appointed to a managerial position. Existing similarity assessment methods do not consider such analogies and do not provide means for pointing them out.

With the goal of spotting context dependent correspondences such as the ones demonstrated above in mind, the present work addresses questions situated one-step ahead of the traditional similarity assessment task: given objects – fragments of news articles in this case – that are already assumed to be similar, *how* they are related to each other? How to identify those aspects of similarity that would facilitate knowledge relevant to both analogized objects?

The research in cognitive science has acknowledged the role of analogy in human thinking. Several works on analogy making have suggested computational mechanisms for constructing detailed mapping that connects corresponding ingredients across a given pair of systems between which analogy is being drawn. According to the *structure mapping* theory (Gentner, 1983), the ingredients of two analogized systems are not expected to share similar individual features, but rather the relations among the ingredients within each system should resemble each other. Another approach (Hofstadter et al., 1995) emphasizes the manner in which features of the analogized objects are perceived in light of the context of aligning them one against the other. Representations that are suitable for mutual mapping are dynamically formed in interaction with the perceived relevance of the features to the correspondence being established. In the next chapter (second part), we review in greater detail these two approaches to analogy making.

The present work is inspired by the direction of constructing a correspondence map, which is grounded on cognitive considerations. We have not found, however, the computational mechanisms employed by cognitive studies directly applicable to real world problems of the type we are interested in, such as identification of corresponding themes in un-annotated texts. One of the above methods (Hofstadter et al., 1995), for instance, has been intrinsically designed for a specific toy problem. This is justified by the authors' claim that studying toy problems is the best strategy for progress in the field. In their view, the current state of our understanding does not allow more realistic models of analogy making. The other approach (Gentner, 1983) is supposed to be applicable to real world problems of general nature, but it represents information about such problems through pre-coded relational representation. It is not clear if and how information embodied in readily available real world data – free text, for example – can be transformed automatically into this type of relational representation.

Eventually, we have coped with the task of identifying context-dependent correspondences across real-world datasets through adapting up to date computational learning methods. Our work thus bridges between cognitive observations regarding analogy making, which inspired our work but provided no concrete utilizable computational recipes, with techniques that have been proven efficient in processing real world data.

Our strategy is unsupervised: we seek to identify patterns or regularities in the given data without the presence of any examples of suitable correspondences. This approach is practically reasonable, as un-annotated data is freely available in many domains and a newly introduced task would require, on the other hand, a considerable amount of work in preparing training data. Further, it is not clear whether the same factors that underlie a particular correspondence would work in other examples. The use of an unsupervised approach thus makes sense also because it might facilitate something of the non-repeating creative nature of the task.

More specifically, our approach extends recent works on data clustering. *Standard data clustering* methods (a term that we shall used interchangeably with *single-set clustering* to distinguish it from our original elaborations) impose structure of the most elementary form on unstructured data by partitioning the given set of data elements into disjoint clusters. In the next chapter (first part), we refer to the data clustering problem in detail and describe methods addressing it.

In our setting, the given element set is pre-divided to several subsets, and our goal is not just to find the immediate internal structure of each of these subsets but rather to find structure that reveals correspondence between them. In particular, we would like to ignore and, in the more interesting cases, to mask out actively internal structures that are irrelevant to the cross-subset correspondence. To that end, our approach extends the standard clustering task by producing clusters that contain elements of both subsets between which we seek to identify correspondence. Each cluster would thus designate concrete links across ingredients or aspects of systems between which an analogy is drawn. Illustratively, a standard clustering method might cluster together names of employees working for different firms in different clusters. Assuming the employer based partition is pre-given, we would expect our approach to produce clusters that capture, say, corresponding functions: the management teams of all firms would be clustered together, the same for the sales teams and so on (as opposed to clusters that coincide with firm-specific units). We present this conception in more detail, exemplify it and describe how performance on this task will be evaluated in Chapter 3.

The first method that we introduce, termed *coupled clustering*, is designed for a dataset pre-divided to two disjoint subsets (each associated with one of the analogized systems). We study the problem of partitioning the two subsets into corresponding sub-clusters, so that every such sub-cluster is matched with a counterpart in the other subset. This target is accomplished through elaboration on an axiomatic cost-based framework of pairwise (i.e., proximity based) data clustering (Puzicha, Hofmann & Buhmann, 2000). Puzicha et al.'s original framework is reviewed in Chapter 4, followed by several alternative extensions aiming at our task. The various extensions are tested and evaluated on synthetic and textual data.

In chapter 5, we introduce another method, *cross-partition clustering*, modifying and generalizing the coupled clustering setting along several aspects: it is based on vectorial representation of the given data rather than on pairwise proximity values, it produces soft (probabilistic) partitioning rather than deterministic one and it allows revealing correspondences across more than two subsets. The cross-partition clustering method that we have developed is based on the *information bottleneck* (Tishby, Pereira & Bialek, 1999) and *information distortion* (Gedeon, Parker & Dimitrov, 2003) methods, which are grounded on information theoretic approach to data clustering. We review in detail these methods before introducing our original elaborations. The cross partition method is tested, as well, on synthetic and textual data, demonstrating noticeable improvement relatively to the coupled clustering results.

Cross partition clustering is a newly defined computational task of general purpose. Potentially, correspondences of the type we study could be drawn across real world composite objects within unrestricted variety of domains. One might be interested, for instance, in identifying corresponding objects in different images. Further examples are discovery of corresponding biological and psychological phenomena typical to different populations, discovery of corresponding business and moves in competing commercial firms (*competitive intelligence*) and so on. The methods introduced in this work are developed using general formulation that would allow adapting them to any application such as the ones mentioned, given data in standard format: similarities between pairs of data elements (coupled clustering) or probabilistic vectorial representation (cross partition clustering).

In accordance with our concrete illustrative examples, the focus of the experimental part of this work is on textual data, specifically, identifying corresponding sub-topics across related, but distinct, domains (rather than short articles or text fragments as in the previous examples). Each domain is represented by a set of keywords extracted from a corpus of texts discussing it. A keyword is characterized by a vector of its co-occurrences with other words in the corpus. Such co-occurrence based representation, which utilizes the fact that similar words are used in similar lexical contexts, is commonly used for tasks such as estimation of similarity between words and documents. In this work, we utilize the correspondences in context captured by the co-occurrence statistics in order to identify correspondences between groups of words.

More concretely, in Chapter 4, we demonstrate how the coupled clustering method performs on identifying correspondences between conflicts of different nature that were discussed extensively in the new articles. The main body of experimental work, in both Chapters 4 and 5, is concentrated on identifying correspondence between different religions: Buddhism, Christianity, Hinduism, Islam and Judaism. Each of these religions is represented by a collection of texts discussing it. The results (due to both coupled clustering and cross partition clustering) are systematically evaluated against clusters manually produced by experts in the comparative study of religions. Some of our results, particularly those in Chapter 5, reveal fundamental themes common to all religions, which can also be traced in comparative studies on religious.

The computational methods introduced in this work provide novel perspective into the essence of analogies and deep semantic correspondences (as opposed to immediate correspondence, based on superficial appearance). This and further aspects and potential elaborations of our work are discussed in the concluding discussion chapter.

Chapter 2: Background

The current work massively builds on the notion of *data clustering*. This fundamental task has been a subject for extensive study of computational disciplines such as machine learning and pattern recognition and is useful for a variety of applications in many empirical domains. The review below intends to provide a broad perspective on the methods that are elaborated later on. The original parts of our work introduced in later chapters elaborate on two particular data clustering methods, which are described in more detail therein. Chapter 4 below is based on Puzicha, Hofmann & Buhmann's (2000) *theory of proximity based clustering*. Chapter 5 is based on two closely related data clustering methods: the *information bottleneck* method (IB; Tishby, Pereira & Bialek, 1999) and the *information distortion* method (ID, Gedeon, Parker & Dimitrov, 2003; Pereira, Tishby & Lee, 1993).

This work is largely inspired as well, though in a less technical level, by computational models of analogy – a field of study in cognitive science, on the borderline between linguistics, psychology and the computational sciences. This chapter therefore concludes with an exemplifying discussion of this field.

2.1 Data Clustering

Data clustering is a computational task, which aims at revealing structure in initially unstructured data. The following definition by Aldenderfer & Blashfield (1984),

The segmentation of heterogeneous set of elements into a collection of homogenous subsets,

provides a good notion of what data clustering is about.

2.1.1 Introduction

The ability to cluster data might be useful for the functioning of intelligent systems, whether artificial or natural. Two closely related motivations for why intelligent systems are expected to develop data-clustering capabilities are (after MacKay, 2003, Ch. 20):

- **Prediction**: characterizing newly encountered pieces of information as exemplars of identifiable distinct classes.
- **Communication**: by referring to similar pieces of information by a common cluster label, communicating parties can concentrate on what is important to the communicated information, while avoiding unneeded details and subtleties.

The above two considerations are in fact closely related. For example, we might expect an intelligent system encountering an object to be able to identify it as, say, ‘an animal’, ‘a plant’, or ‘a piece of furniture’ and, at a finer level of resolution, as ‘a cat’ or ‘a tiger’. Such labeling system might play a role in classifying data both for the system's own purposes and for communicating this information to other parties.

Data clustering methods approach the above restrictedly from unsupervised perspective. Likewise, the computational mechanisms reviewed and developed throughout this work rely on unlabeled training data, while external supervision providing known-to-be “correct labels” is not introduced during learning (as is indeed the case in many practical settings).

2.1.1.1 Practical Applications

There are many applicative uses for data clustering. A partial illustrative list is as follows:

- **Object recognition**: partitioning of a set of images, recorded sounds, or other composite records that are not subject to immediate automated interpretation, into well-distinguished classes. The intended scope of each “well-distinguished class” may vary. Few examples are:
    - all exemplars of images from the same natural category (e.g., “animals” in one cluster, “buildings” in another one and “pieces of furniture” in a third cluster).
    - all exemplars of the same object photographed under different conditions (changing illumination, angle etc.)
    - all exemplars of recorded pronunciations of the same word (e.g., names of alphabet letters, as in Blatt, Wiseman, & Domany, 1997)
- **Image segmentation**: the classifications of image pixels to different textures or to different objects (Hofmann & Puzicha, 1998; Boykov, Vexler & Zabih, 1999), e.g. objects in a room or organs in an image produced by some medical imaging technique.
- **Information Retrieval and Natural Language Processing**: topical categorization of documents (Dhillon & Modha, 2001; Slonim & Tishby, 2002); detecting senses and sub-topics through word clusters (Pereira, Tishby & Lee, 1993; Brew & Schulte im Walde, 2002; Korhonen, Krymolowski & Marx, 2003).

Forming clusters of semantically similar documents or words is in particular related with the applicative part of the current work. Word clusters illustrate how what can be called “a conceptual network”, where each cluster of words sharing a common sense reveals a different concept, emerges from unprocessed textual data. This is accomplished with no supervision, not to mention “deep understanding” of the language the analyzed texts are written in. In our case, as well as in other works such as the two above references on document clustering, clusters are produced with no information about the texts except word co-occurrence statistics (the three works on word clustering cited above use syntactic information extracted automatically prior to the clustering). Thus, beyond its applicative utility data clustering can be seen as demonstrating, on a very basic level, computation-based intelligence and the emergence of meaning.

2.1.1.2 Data Clustering is an Ill-posed Task

A well-known fact about data clustering is that it is *ill posed*, or, as Estivill-Castro (2003) puts it: “the cluster is in the eyes of the beholder”. Any definition to the data-clustering problem does not set an unambiguous criterion to judging whether, or how well, the problem is solved. Quite often, several partitions exist representing compromises between different biases inherent to the data, so the user must be more detailed with regard to the specific goals and the considerations relevant to those goals. For instance, image segmentation data is constrained not only by pixels' color or grey scale, but also by the spatial proximity of the clustered pixels. Spatial proximity, however, is irrelevant in most other settings. Being more specific and posing more restrictions does not necessarily turns the data clustering into a well-defined problem. Assuming an axiomatic framework that poses some more concrete restrictions, Kleinberg (2003; see Subsection 2.1.4.3 below) shows that the data clustering problem is inherently unsatisfiable.

2.1.2 The Structure of Data Clustering Output

Basically, a data clustering method is supposed, given a set of elements, to produce a partition of the set into clusters. We use the term *clustering configuration* to denote any one of all possible partitions of the data, among which data clustering methods are supposed to identify the one that optimally addresses the data-clustering task. Each cluster is, by definition, just the set of data elements that forms it. There are, however, methods that provide also supplementary information. There are several methods that specify also a *prototypical representation* for every cluster, which gives an idea about each cluster's characteristics while saving the need to examine its elements. We elaborate on such cases of extended clustering output later on (see 2.1.4.4). In this subsection, we discuss two issues that are part of the clustering output in its more basic form: the number of clusters being produced and multi assignments and probabilistic assignments of data elements to clusters.

2.1.2.1 How Many Clusters

It is quite clear that the number of clusters, *k*, is expected to be considerably less than the number of clustered elements. However, determining *k* exactly is a non-trivial issue, which might depend, for instance, on specific user requirements. There are methods (e.g., Blatt, Wiseman & Domany, 1997) that infer *k* from the data itself, with no directions regarding user preferences. There are also statistical significance tests comparing configurations of k versus k+1 clusters in order to judge whether producing the larger number of clusters is justified (Duda, Hart & Strock, 2001, pp. 557-559). Many methods, however, including those adapted in the current work, follow user instructions. In these cases, k is usually specified as an additional input parameter. Alternatively, some methods require the user to indicate the value of a method-specific threshold parameter, which indirectly determines k, with dependence on the data.

2.1.2.2 Assignment Probabilities

In many situations, partitioning the data to homogeneous subsets leaves some elements that do not fit perfectly into one of the clusters. One conceivable solution, sometimes termed "soft" or non-deterministic clustering, is to allow the assignment of an element to two or more clusters, with varying degrees of justification or confidence. Hence, the clustering output of several methods includes detailed information with regard to "level of assignment" $ass(c,x)$ of each element x within each cluster c. A common convention is to restrict the assignment levels to be non-negative, $ass(c,x) \geq 0$, and to require all assignment levels of an individual element x to sum up to 1: $\sum_c ass(c,x) = 1$. Under this convention, the level by which an element is assigned to a cluster can be interpreted as $ass(c,x) \equiv p(c|x)$, the *probability* of the (deterministic) assignment to c to take place, given that the assigned element is x. We call methods with this kind of output *probabilistic clustering* methods. In this work, we use a probabilistic framework in Chapter 5.

2.1.3 Data Representation

Different data clustering methods differ by the type of data representation they are capable of dealing with – the format of the input they can take.

2.1.3.1 Pairwise Representation

Pairwise clustering methods rely on measures of proximity, i.e. similarity or dissimilarity values, between pairs of data elements. An example for a direct source for similarity values utilizable for pairwise clustering is a confusion matrix (e.g., Manos, 1996). This is a matrix, based on the performance of subjects under study, in which both rows and columns correspond to data elements. The entry at a row corresponding to an element x and column corresponding to an element x' indicates empirical count $count(x \leftarrow x')$, or estimated probability $p(x|x')$, of miss-recognizing an element x' as an element x. More on proximity measures in 2.1.3.3 below.

2.1.3.2 Feature-based Representation

Somewhat more typical to actual data than pairwise representation is a representation where, along with the set of elements to be clustered, an auxiliary set of features that are considered relevant to the desired outcome (*relevance features*) is present. In such case, each element is identifiable as a vector with entries that reflect how intensive the element's association with each of the various features is. For example, an element *x* can be represented by the vector of observed co-occurrence counts of *x* with each one of the features *y*. Such co-occurrence vectors are widely used, including in this work.

There are clustering methods that process co-occurrence vectors in their normalized form, i.e., vectors of estimated conditional probabilities $p(y|x)$ of each feature *y* to co-occur with an element *x*. The normalization factor is the total occurrence count of the element *x* in the data, over all features with which *x* co-occurs ($count(x)$). In a probabilistic setting, the total counts of data elements are often normalized themselves to a probabilistic vector of relative frequencies $p(x)$, for each data element *x*.

2.1.3.3 Proximity Measures

Similarity and dissimilarity assessment are practiced both as an independent unsupervised task and as pre-processing for other tasks, including data clustering. It is used within many applications: data mining (Das, Mannila & Ronkainen, 1998), image retrieval (Ortega et al., 1998) and document clustering (Dhillon & Modha, 2001).

Dissimilarity can be viewed as distance in the feature space. The well-known L_1 (*Manhattan*) norm, which is the sum of absolute differences between corresponding vector coordinates

$$L_1(x,x') \;=\; \sum_i |x_i - x'_i| \;, \tag{2.1}$$

where corresponding coordinates are indexed by a common index *i*, and the L_2 (*Euclidean*) *distance*, which is the square root of sum of squared coordinate differences

$$L_2(x,x') \;=\; \sqrt{\sum_i (x_i - x'_i)^2} \;, \tag{2.2}$$

are sometimes used as benchmarks (e.g. by Lee, 1999). In general, there are no strict formal restrictions on the similarity or distance values. For instance, a dissimilarity measure is not expected to form a *metric* as L_1 and L_2 do, and in particular, a distance measure *d* is not required to be symmetric ($d(x,x') \neq d(x',x)$ is permitted) or to obey the triangle inequality ($d(x,x') + d(x',x'') \leq d(x,x'')$ is permitted).

Several dissimilarity measures refer to representation of data in the above mentioned form of conditional probability distributions of features. The *Kullback-Leibler* (*KL*) *divergence* (Cover & Thomas, 1991, p. 18) quantifies the inefficiency of coding information distributed according to $p(y|x')$ with code that is optimized for $p(y|x)$:

$$KL[p(y|x)||p(y|x')] = \sum_y p(y|x)\,(\,\log p(y|x) - \log p(y|x')\,). \tag{2.3}$$

KL divergence is undefined, i.e. approaches infinity, whenever there is a feature y^* such that $p(y^*|x) > 0$ but $p(y^*|x') = 0$. There are dissimilarity measures that are based on *KL* divergence but overcome this problem. The *Jensen-Shannon* (*JS*) *divergence* (used, e.g., in Manos, 1996, and Korhonen, Krymolowski & Marx, 2003) is the sum of *KL* divergences of $p(y|x)$ and $p(y|x')$ from their average, possibly weighted by the elements' relative frequencies $p(x)$ and $p(x')$:

$$JS[p(y|x)||p(y|x')] = \pi KL[p(y|x)||q(y)] + \pi' KL[p(y|x')||q(y)]\,, \qquad (2.4)$$

where $\pi = p(x) \,/\, (\,p(x) + p(x')\,)$, $\pi' = p(x') \,/\, (\,p(x) + p(x')\,)$ and $q(y) = (\,\pi p(y|x) + \pi' p(y|x')\,)$. The *ff-skew divergence* (Lee, 1999) is the *KL* divergence between $p(y|x)$ and a slight shift, by a small positive value α, of $p(y|x')$ towards $p(y|x)$:

$$ff\text{-}skew[p(y|x)||p(y|x')] = KL[p(y|x)\,||\,(1-\alpha)p(y|x') + \alpha p(y|x)]\,. \qquad (2.5)$$

Both the symmetric *JS* divergence and the non-symmetric *ff*-skew divergence are guaranteed not to approach infinity, because the average of $p(y|x)$ and $p(y|x')$, as well as the shifted probability p', are equal to zero only on those features y for which both $p(y|x)$ and $p(y|x')$ are equal to zero.

Similarity values are often induced from distance values. A popular scheme of calculating similarity between two data elements x and x' is through an exponentially decreasing function of their given distance d: $sim(x,x') = e^{-d(x,x')}$ (as, e.g., done by Blatt, Wiseman & Domany, 1997). This scheme tends to sharpen differences between pairs of close neighbors, while blurring differences between pairs of distant elements, thus it intensifies sensitivity to neighborhood-related information that is more relevant to the clustering task (distant elements would always be assigned to different clusters).

In the domain of text processing, co-occurrence based similarity measures are widely used (see Dagan, 2000 for a review). Such measures rely on the observation that semantically similar words tend to share similar patterns of co-occurrences with their neighboring words, which take the role of relevance features in this case. Likewise, similar documents share similar word frequency distributions. A common convention bounds the similarity values between 0 to 1, where 1 typically denotes self-similarity. One widely used measure is the cosine of the angel between two co-occurrence count vectors (van Rijsbergen, 1979, ch. 5; used, e.g., by Dhillon & Modha, 2001), that is, the sum of corresponding entry products (dot product) of the two vectors normalized by the L_2 norm of the product:

$$cosine\,(x,x') \;=\; \frac{\sum_y count(x,y)count(x',y)}{\sqrt{\sum_y count(x,y)^2}\sqrt{\sum_y count(x',y)^2}}\;, \qquad (2.6)$$

The cosine and other similarly straightforward measures are affected by the data sparseness problem, intensified, for instance, by the common use of different words referring to similar meanings. More sophisticated measures, which are designed to overcome the sparseness problem, incorporate information-theoretic perspective. We employ in this work two such similarity measures, by Lin (1998) and Dagan, Marcus & Markovitch (1995). These measures are described in detail later, in Chapter 4.

2.1.3.4 Re-representation and Preprocessing

There are cases where the given representation of the data is modified before applying the actual clustering method. This might be required, for instance, because the method in use takes different representation than the given one, or in order to have smaller or less noisy data so that the method operates faster or produces results of better quality.

Re-representation in general implies loss of information. Consequently, an important sub-goal of pre-processing is to preserve the information relevant to the clustering task. When pairwise data is given, similarities between distant elements are often ignored, resulting in sparse similarity matrix, which improves computation efficiency. This is an example of rather straightforward elimination of irrelevant information: as noted, any two unmistakably dissimilar elements are expected to be in different clusters and their exact level of similarity is less important.

A common re-representation procedure is the transformation of feature vectors into an array of similarities, or distances, appropriate for pairwise clustering. Such transformation is based on a measure of distance or similarity between vectors (see previous subsection). Often, terms such as 'similarity' or 'distance' are mentioned also in the context of feature-based clustering, which gets a concrete meaning only if a concrete measure of similarity between feature vectors is specified.

Feature-selection and feature-generation techniques can be applied to convert one form of vectorial representation into another, with aims such as masking noise from the given representation or decreasing the data complexity prior to actual clustering. For example, PCA (*principled component analysis),* a dimensionality-reduction procedure, is used by Brew & Schulte im Walde (2002) as a pre-processing procedure prior to applying a clustering method.

2.1.4 Algorithmic Framework for Data Clustering

The number of all possible partitions of n elements to k clusters is roughly $k^n/k!$, which grows exponentially with n (Duda, Hart & Stork, 2001, p. 548). Hence, going through all different clustering configurations in search for one realizing the requirements, whatever they are, is computationally ineffective (indeed, data clustering is an NP-complete problem, Garey & Johnson, 1979). Accordingly, clustering algorithms typically attempt to provide, by means of restricted but principled search, a reasonably good solution even if spotting the absolutely best solution in not guaranteed. The following subsections present some schemes for how various data clustering methods conduct such a search.

2.1.4.1 Incremental Search

A strategy for solving optimization problems that are not liable to an exhaustive search, implemented within most data clustering methods that are mentioned below, is performing the task incrementally, as shown in Fig 2.1.

```
given an initial configuration of clusters
repeat
      from a set of currently available update steps of a pre-specified
      type, perform the one that is optimal due to some pre-specified
      criterion, so that a new clustering configuration results
Until the resulting configuration meets some pre-specified stop condition
```

Figure 2.1: The incremental clustering scheme.

There are several simple examples for methods implementing this step-by-step scheme. One well-known class of such methods assumes an initial configuration where each element forms an individual cluster (*singleton*) and the update steps are cluster merges. This strategy, known as *agglomerative clustering*, elementarily produces a strict hierarchy of clustering configurations. Criteria to assess the best merge to perform are, for instance, the similarity of most similar members of the clusters to be merged, or the similarity of their most dissimilar members (known, respectively, as the *single linkage* and *complete linkage* methods, Duda, Hart & Stork, 2001, pp. 553-554). The stop condition in these examples is either the formation of a pre-specified number of clusters *k* or a pre-specified similarity threshold value beyond which clusters are not merged any more.

Other feasible types of update steps employed by different methods, other than cluster merges, are: cluster splits, reassignments of a single data element and reassignment of all elements. Identifying the optimal update step, which means quantifying each steps' quality relatively to the other candidate steps, should be carried out with low computational complexity in order to maintain tractability.

The stepwise scheme, though prevalent, is not the only conceivable search strategy. Blatt, Wiseman & Domany (1997), for instance, implement a Monte Carlo procedure producing a series of partitions that can be thought of as “typical” rather than “target” clustering configurations. Then, the probability of element pairs to be part of the same typical cluster is used to determine the final configuration. (Formally, however, this process can also be understood as a conversion – or re-representation, see 2.1.3.3 above – of the given similarities into similarities of other type, namely probabilities to share the same “typical” cluster, followed by application of the single-linkage method with threshold 0.5).

2.1.4.2 Cost-Based Search

There are data clustering methods that assign to each admissible clustering-configuration a measure of quality, or a *cost function*. The availability of global quality measure or cost function has several attractive consequences. The target of data clustering becomes clear and concrete: finding a configuration of highest quality or lowest cost. Accordingly, the stepwise scheme described in the previous subsection can be made more concrete given a cost function, so that the criteria of how to choose the update step to execute next and when to stop the iterative loop are clear and explicit:

```
Given an initial configuration of clusters
repeat
      from a set of currently available update steps of a pre-specified
      type, perform the one reducing a pre-specified cost more than any
      other step, so that a new clustering configuration results
Until the cost cannot be reduced by any available step
```

Figure 2.2: The cost-reduction clustering scheme.

The above iterative loop is guaranteed to stop, as the cost is bounded from below (the number of configurations is finite) and is reduced every iteration. It is apparent, however, that the obtained configuration is not necessarily of absolutely lowest cost, but just one that cannot be improved by the available updates steps. Update step types usually employed are single element re-assignment, picked either according to some fixed schedule or at random (Puzicha, Hofmann & Buhmann, 2000; Slonim, Friedman & Tishby, 2002). In the last case, before terminating the iterative loop, it should be verified that cost cannot improve by any further reassignments, not just of the element currently examined.

2.1.4.3 Axiomatic Approach to Data Clustering

An attractive aspect of cost functions is that they provide a definite criterion, in terms of precise mathematical expression, for determining the quality of data clustering outcome relatively to other possible outcomes. However, in order to make a rough definition of the data clustering problem (such as the one above at the top of Section 2.1) precise, a cost function must involve further assumptions that are not necessarily consensual, but rather they reflect more individual view of what should be expected of “a reasonable clustering procedure”. Assumptions of this kind are sometimes incorporated within an axiomatic approach to data clustering. We exemplify below two cases where such assumptions are incorporated in the pairwise clustering setting.

Kleinberg (2003) includes a requirement termed *consistency* in his axiomatic analysis. This requirement states that in any specific clustering problem the best (lowest-cost) clustering configuration should remain the best solution also for data that is modified relatively to the original problem as follows: some or all of the similarities between elements sharing the same cluster in the best configuration are increased and between-cluster similarities are decreased. Puzicha, Hofmann & Buhmann's (2000) theory of proximity-based clustering (on which we elaborate in Chapter 4; see detailed review there) introduces other requirements. For instance, the relative ranking of all clustering configurations is not expected, under their assumptions, to change in case all pairwise input values are multiplied or shifted by a constant (*scale* and *shift invariance* properties, respectively).

2.1.4.4 Prototypical Representatives of Clusters

Some clustering algorithms keep prototypical vectorial representation of each one of the clusters, additional to the list of cluster members (as mentioned before, such representations can be understood as a part of the algorithm output).

The incremental data clustering scheme (Figure 2.1) can make use of prototypical representations. When they are incorporated, assessing the candidate update steps can rely on associations between elements and cluster representatives, which are fewer than the associations between all pairs of elements involved. On the other hand, the incremental scheme would require in that case updating the cluster representatives along with the updates of assignments into clusters:

```
given an initial array of k cluster representatives
repeat
      -  reassign all elements - each element to the cluster with the
         representative to which it is most similar
      -  recalculate cluster representatives, based on the members of each
         cluster
until a stable configuration is obtained
```

Figure 2.3: The *k*-representatives clustering scheme.

The *k*-representatives scheme is a specific heuristic variant of the incremental scheme. The specific stopping condition, namely obtaining a stable configuration where cluster representatives are not liable to any further change, is not guaranteed in general terms. It turns out, however, that in some concrete cases, with certain definitions of cluster representatives and element-representative similarity relations, the *k*-representatives scheme happens to reduce a specific cost criterion (known as the *Lyapunov function* of the algorithm, McKay, 2003, p. 299). Therefore, such cases, three of which are mentioned below, realize also the cost-based clustering scheme (Figure 2.2) and are hence assured to stop with configuration that locally minimizes the specific cost.

Feature-based methods naturally represent clusters as vectors in the same space of their input vectorial representations:

- **The *k-means* algorithm**: Each cluster is represented by its *centroid* – the mean of the vectorial representations of the cluster's elements. The distance (dissimilarity) between the centroids and the data elements is measured in L_2 (Euclidean) norm. In this case, the cost being reduced happens to be the sum of squared L_2 distances of all data elements, each from its cluster centroid (Estivill-Castro, 2003).

Duda, Hart & Stork (2001, p. 550) note that, in terms of susceptibility to being trapped in local minima, the *k*-means algorithm has been found empirically advantageous over cost based search with reassignment of a single element at a time.

- **The *k-medoids* algorithm**: Each cluster is represented by its *medoid* – a vector where each entry is the median value of the corresponding cluster-member entries. Element-medoid similarity is captured through proximity in L_1 norm. The cost being reduced in this case is the sum of L_1 distances between all elements and their cluster medoids (Estivill-Castro, 2003).

The notion of cluster representative is applicable, though not very intuitive or common, also in pairwise clustering:

- The prototypical representative is a concrete data element, for which the sum of similarities to all other members of its cluster is the largest compared to corresponding sums of the other cluster's members (equivalently, the sum of dissimilarities is required to be the smallest in the cluster). The cluster-element similarity or dissimilarity employed is straightforwardly given by the proximity measure between data elements. The cost being reduced in this case is the sum of dissimilarities, or minus the sum of similarities, between all the elements and their corresponding cluster representatives. (It is easy to verify that, in each iteration, the total cost is decreased by both reassignments and re-selected representatives).

2.1.4.5 Stochasticity

Randomness can affect several aspects of the algorithmic schemes introduced so far. For instance, the initial configuration can be determined randomly and so is the next element to be reassigned, in case a single element is reassigned at each update step of the incremental scheme. More importantly, update steps can be chosen stochastically among all alternatives as the following scheme explicates:

```
given an initial configuration of clusters
repeat
      pick at random one of the currently available update steps $s$ and
      perform it with probability correlated with the anticipated addition
      to cost $\Delta s$
until, for some pre-specified number of iterations, the change in cost
does not exceed some pre-specified small threshold value
```

Figure 2.4: The stochastic cost-driven clustering scheme.

There are a couple of algorithmic variations that fit this scheme. They differ from one another by the way they calculate the probability to perform a candidate step as a function of its anticipated impact on the cost. As with the basic cost-reduction scheme, the update steps in both variations are picked from all possible reassignments of one randomly chosen element. In one variation of *simulated annealing* (Kirkpatrick, Gelatt & Vecchi, 1983) if an update step that does not increase the cost is picked it would be performed always (i.e. in probability 1). A step s that does increase the cost by a positive quantity Δs would be performed in probability $e^{-\beta\Delta s}$. In another variation (*Gibbs sampling*, Geman & Geman, 1984; implemented in Chapter 4) any step s can be picked and performed in probability proportional to $e^{-\beta\Delta s}$, where the cost change Δs can be either positive or not. In both cases, the cost might occasionally grow, though rarely in comparison to cost reduction. Stochasticity is gradually relaxed during execution of both variations by means of gradual increasing of the inverse "computational temperature" parameter β. Initially, β is low, which implies low differentiation between varying levels of cost-change. During execution β is gradually increased, so that after a large number of iterations a high β value is employed and systematic increase in cost becomes very probable even in comparison to slight deterioration.

Theoretically, and often in practice, in order to ensure that the algorithms above produce a clustering configuration of globally low cost, a very slow schedule of stochasticity relaxation (i.e., very gradual increase of the β parameter) is required, which results in long execution time (Duda, Hart & Stork, 2001, p. 356).

2.1.4.6 Probabilistic Clustering

As mentioned before (in Subsection 2.1.2.2), there are methods that compute non-negative probabilities of assignments or "assignment levels" $p(c|x)$, which, per element x, sum up to one over all clusters c. We will now refer to those methods assuming further that the given feature-based vectorial representations of the elements are also normalized: each element x is represented by conditional probability distribution $p(y|x)$ over the features. Given such normalized representation of the data, it is natural that cluster representatives are taken from the same probabilistic space: each cluster is represented by a conditional probability distribution over the features, $p(y|c)$. This probabilistic setting underlies the following scheme, which generalizes the *k*-representatives clustering scheme of Figure 2.3:

```
given initial assignment levels $p(c|x)$ for each element $x$ and cluster $c$,
repeat
    -   recalculate each cluster probabilistic representative $p(y|c)$, as a
        normalized sum of all element probabilistic representations $p(y|x)$,
        weighing the contribution of each element $x$ by its assignment
        probability to $c$, $p(c|x)$.
    -   recalculate each assignment probability $p(c|x)$ for each element $x$ and
        cluster $c$, so that it is proportional to the similarity of $x$'s
        vectorial representation, $p(y|x)$, to $c$'s vectorial representation,
        $p(y|c)$.
until the recalculated representatives do not change any more beyond some
pre-specified small threshold value
```

Figure 2.5: probabilistic representative-based clustering scheme.

We refer below to two approaches that fit in the above probabilistic representative-based scheme. These two approaches, whose underlying algorithms are very similar to one other, are related with the two motivations mentioned earlier for the data clustering task: "prediction" versus "communication" (Subsection 2.1.1). One approach – the one associated with the predictive aspect of data clustering – follows the *expectation maximization* (EM; Dempster, Laird & Rubin, 1977) framework. It examines the data as if it were sampled from a mixture of latent distinguishable classes of element-feature co-occurrence distributions to be approximated. The iterative steps of calculating assignment probabilities $p(c|x)$ and cluster representatives $p(y|c)$ maximize a likelihood term of the mixture model. Deriving the update steps so that they systematically maximize the model likelihood implies a particular representative-element similarity measure,

$$sim_{EM}(x, c) = p(c)\, e^{-count(x)\, KL[\, p(y|x) \,||\, p(y|c)\,]}, \qquad (2.7)$$

where $p(c)$ is the relative weight of cluster c and $count(x)$ is the total number of occurrences of the element x in the data. A particular work that is based on this formulation is Hofmann, Puzicha & Jordan's (1999) *one-sided clustering model*. As the data – i.e., the number of times each element is sampled – grows, the method is capable of assigning the elements more deterministically and detecting larger numbers of clusters.

The second approach is related with the communication aspect of clustering and is based on information theoretical considerations. The *information bottleneck* method (IB; Tishby, Pereira & Bialek, 1999) implements this approach. It looks at data clustering as if it aims at lossy encoding of the data so that relevant information, namely information about the features, is conveyed optimally. In this case as well, a similarity measure emerges from the principles underlying the IB method, which turns to be very similar to the EM related similarity measure:

$$sim_{IB}(x, c) = p(c)\, e^{-\beta KL[p(y|x) \,||\, p(y|c)]}, \tag{2.8}$$

where $p(c)$ is, again, the relative weight of the cluster c and β is an additional parameter, which, roughly speaking, articulates counterbalance between the target of communicating the information as accurately as possible and the target of reducing the length of communicated transmission. In Chapter 5, we provide more detailed description of the IB method and the closely related *information distortion* method (Gedeon, Parker & Dimitrov, 2003; Pereira, Tishby & Lee, 1993).

Further examples of methods following the probabilistic representative-based clustering scheme can be seen as close variants on the methods mentioned above, for instance, the three *soft k-means* versions that are specified by MacKay (2003, Ch. 20, 22). Another method that is worth mentioning in this context is Bezdek's (1981) *fuzzy c-means*, which does not require the assignment levels of an element to sum up over all clusters to one, but leaves space for uncertainty regarding the assignments.

2.1.5 Variations on Data-Clustering

We now discuss some methods that are related to, or extend, the data clustering task.

2.1.5.1 Data Clustering and other Unsupervised Tasks

Data clustering is a key member in the family of unsupervised computational learning methods. It is interesting to note that data clustering can be seen as if it accomplishes tasks that we have already mentioned as means for pre-processing prior to clustering (Subsection 2.1.3.4).

- **Similarity assessment**: any standard deterministic data-clustering configuration imposes a trivial 0/1 similarity measure: elements of the same clusters are considered similar, while elements of different clusters are not. Probabilistic clustering is more detailed in this respect: assignment probability distributions of individual elements over the clusters, $p(c|x)$, can be used to measure distance or similarity between data elements through standard distance or similarity measures of distributions (e.g., *KL* distance). This might facilitate overcoming the sparseness that often characterizes raw element-feature co-occurrence vectors in domains such as text processing.

- **Dimensional reduction / feature generation**: clustering is also a particular case of dimensional reduction. Specifically, probabilistic clustering maps the data onto a *k*-1 dimensional simplex embedded in a *k* dimensional space, where *k* is the number of clusters. The utility of dimensional reduction of the feature space – replacing the original array of features by clusters of features – is demonstrated by Slonim & Tishby, 2000.

2.1.5.2 Methods that Extend Basic Data Clustering

There are data clustering methods, both deterministic and probabilistic, that process and output constructs more elaborated than plain clustering configurations. There are several methods that output a *hierarchy* of clusters, i.e., a sequence of increasingly detailed clustering configurations consisting of an increasing number of smaller clusters, so that every cluster forms a subset of a cluster in each one of all less detailed configurations. Simple examples are the complete and single linkage methods mentioned before (Subsection 2.1.4.1). Hierarchical clustering, however, can be tackled through more sophisticated approach, including the case of probabilistic clustering (Hofmann, Puzicha & Jordan, 1999). Although in the current work we do not aim at hierarchy in general, we note that in general meaningful sub-clusters that emerge as individual clusters in a more detailed configuration are often revealed simply by applying a non-hierarchical method repeatedly with number of clusters, *k*, incremented at each run. Such partial hierarchy often reveals interesting relations between themes and sub-themes in the data (as demonstrated in Chapter 4 and Chapter 5).

One further well known modification to the data-clustering task is the *self-organizing map* (SOM; Kohonen, 1989) that, in addition to a clustering configuration, maps the clusters onto a grid, thus imposes spatial structure on the clusters.

A recent line of works on dimensionality reduction (the *two-sided clustering model* by Hofmann, Puzicha & Jordan, 1999; Hofmann, 1999; Globerzon & Tishby, 2002), can be seen as extending feature-based clustering to a setting of two sets of elements, symmetrically playing the roles of both clustered data and features with respect to each other. A further recent work, on the *multivariate information bottleneck* method (Friedman et al., 2002), extends the concept and technique of the IB method in revealing complex relational constructs. The input to this method may consist of several distinct element sets that are connected with one another through a network of element-feature relations. The method can produce several clustering configurations, each of which partitions (probabilistically) one of the sets that take the role of clustered data. The obtained complex relational structure allows, for example, several different partitions of the same set simultaneously, directed by different relevance feature sets conveying different types of information regarding the "multi-clustered" set.

2.1.5.3 Data Clustering with Constraints

There are several works, including the present one, on methods producing output that has the form of a standard clustering configuration, but differ from ordinary data clustering by considering input that is supplementary to the standard array of pairwise proximity values or feature-based vectorial representations.

In a review of relational data-clustering methods used within social sciences, Batagelj & Ferligoj (2000) provide examples of combining clustering with relational biases of various kinds that are embedded within the data. The *blockmodeling* method seeks to cluster together elements that have similar patterns of relations with other elements (in this case, some representation of the global relationships between clusters may form supplementary output). There are several types of relational pattern similarity, such as *structural equivalence*, where elements are identically related with the rest of individual elements, and *regular equivalence*, where elements are similarly connected to equivalent other elements. Another approach reviewed by Batagelj & Ferligoj (2000) – *constrained clustering* – groups similar elements into clusters based on features, but clusters have to satisfy also some additional conditions. For example: clusters of geographical regions that are similar according to their socioeconomic development level have to be determined such that the regions inside each cluster are also geographically connected. Considerations that, in a way, are similar to the above are applied in works on image segmentation, where nearby pixels are relatively probable to be part of the same object (Boykov, Vexler & Zabih, 1999).

Other works (Wagstaff et al. 2001; Basu, Banerjee & Mooney, 2002) tackle data clustering constrained by other types of pre-specified restrictions. They take as an additional input a list of element pairs constrained to be in the same cluster, versus another list of pairs constrained to be in different clusters. Several variants of the k-means algorithm that incorporate such constraints were suggested.

The method of *information bottleneck with side information* (Chechik & Tishby, 2003) can be viewed as extending further the above line. Chechik & Tishby consider an additional set of features conveying "negative" information that is supposed to be neutralized rather than be followed in forming the clusters. Our work addresses a closely related task, so we will provide detailed comparison with this work.

2.2 Computational Models of Analogy

Analogy is defined as “similarity in some respects between things that are otherwise dissimilar” (quoting http://dictionary.reference.com/search?q=analogy). The issues of how analogies are identified and what makes an analogy a good one have been discussed in the cognitive literature from a variety of points of view. A major motivation to studying computational methods for identifying analogies (*analogy making* in short) is that the capacity of drawing analogies and metaphors is an essential part of human intelligence. Analogy making allows utilizing knowledge, ideas and inspiration across seemingly different domains and thus it contributes significantly to the flexibility and creativity characterizing human intelligence. *“Analogy pervades all our thinking, our everyday speech and our trivial conclusions as well as artistic ways of expression and the highest scientific achievements”* (Polya, 1957).

In addition to the theoretic motivation of modeling an essential ingredient of human intelligence, developments in the computational methods used for analogy making – including, we hope, the ones introduced in this work – might have impact on practical applications: language understanding and generation, data mining, artificial intelligence and so on.

Our review below is only exemplifying. We concentrate on two approaches to analogy making: the *structure mapping theory* (Gentner, 1983) and the *Copycat* project (Hofstadter et al., 1995, Chapters 5-6). Among the many works to which our review does not refer, there are several that can be seen as lying somewhere in between the two above mentioned ones (e.g., Holyoak & Thagard, 1989; Hummel & Holyoak, 1997; Veale, O'Donoghue & Keane, 1999; Kokinov & Petrov 2001)[1]. More comprehensive discussion with reference to a larger variety of works is given, e.g., by French (2002).

2.2.1 The Structure Mapping Theory

As French (2002) notes, Gentner's structure mapping theory (SMT; 1983) is unquestionably the most influential work to date on the modeling of analogy-making. It has been applied in a wide range of contexts ranging from child development to folk physics. The prominent innovation of SMT relatively to earlier works is the emphasis that it puts on structural similarity between the analogized

[1] All in all, these works process data consisting of relational prepositions similar to the representation used by the computational implementation of the structure mapping theory (see Section 2.2.1), but they employ computational machinery, such as connectionist or neural network architectures, somewhat closer in spirit to Copycat (2.2.2).

systems. The *structure-mapping engine* (SME; Falkenhainer, Forbus & Gentner, 1989) is the computational implementation of SMT.

2.2.1.1 Data Representation

SME represents the information about the systems between which analogy is to be drawn as relational prepositions. Unary relations (of one argument) are equivalent with elementary attributes or features, as is familiar from the conventional feature-based data-clustering setting. A numeric value might be used to quantify the association between an element and its attribute. For example (based on Falkenhainer, Forbus & Gentner, 1989), the fact that 'temperature' is an attribute of coffee, with value *X*, is denoted as:

```
TEMPERATURE(coffee) = X
```

In distinction from elementary features, there are relations that apply to two or more elements and even to other relations, e.g.,

```
GREATER-THAN [ TEMPERATURE(coffee), TEMPERATURE(ice-cube) ]
```

or to any fixed combination of relations and data elements.

2.2.1.2 Principles and Algorithmic Framework

Two major principles underlie SMT:

- the relation-matching principle: good analogies are grounded on mapping of multi-argument relations rather than attributes (unary relations).
- the systematicity principle: mappings of coherent systems of relations (i.e., graphs resulting from compositions of relations) are preferred over mappings of individual relations.

The SME algorithm implements a heuristic search for a cross-system map realizing these principles through four stages:

- Local match construction: find all base-target element pairs that can potentially match, i.e. all possible matches between groups of elements sharing *identical* relations in base and target systems.
- Global map construction: within the formed collection of all possible local matches, identify all maximal consistent sub-collections of matches.
- Candidate inference construction: for each maximal consistent sub-collection of matches, infer additional relations not given originally in the target domain that have

matched relations in the base domain and thus extend the map suggested by the consistent matches.

- Match evaluation: calculate a score for each one of the candidate extended maps incorporating the inferred information, based on local structural measures that are derived from the two SMT principles above.

Although innovative and influential (and maybe because of it), SMT has been extensively criticized, for example by Hofstadter et al. (1995, mainly Ch. 4). Hofstadter et al.'s criticism is particularly focused on the inflexibility inherent to SME. This inflexibility is expressed in the one-to-one mapping scheme that is restricted to mapping of identical – even not similar – relations. Further, the propositional representations that the program manipulates are manually coded and, as such, they articulate pre-determined relations over a pre-determined set of concepts. The SMT pretends to account for real-world analogies. However, it is not clear to what extent it models successfully the flexibility and creativity characterizing human reasoning, particularly if one has in mind *real-world data* in raw form that cannot be suspected as tailored for the problem at hand.

2.2.2 The Copycat Project

The Copycat project by Hofstadter et al. (1995; Chs. 5-6) is one of several projects from the same group (see other chapters there), promoting an approach alternative to the seeming rigidity of SMT. A basic strategic direction of Hofstadter et al. is abandoning the pretension to solve real-world problems of general character, which in their view is a too advanced challenge for the present level of recent research. Rather, they advocate concentrating on specific artificial toy domains. Simple toy problems are claimed to underlie search spaces that are more liable to systematic study (as Hofstadter et al. put it: looking at a problem together with its “hallo” of variant problems; p. 330).

2.2.2.1 System Overall Description

Copycat is a computer program exemplifying the above direction. It is restrictedly designed to answer questions regarding analogy of letter strings transformation, such as:

> “if a string ***abc*** is transformed into ***abd***, what would be the analogously transformed value of a target string ***xyz***”.

The strings ***abc*** and ***abd*** form the *base* domain of the problem and the ***xyz*** string is the given part of the *target* domain. The problem to be solved is completing the target domain by constructing an additional string so the relation between the two target strings, the given one and the constructed one, will be analogous to the relation between the two base strings.

The solution is achieved based on grouping letter subsets together and on two types of links between the original letters and between formed letter groups: *bonds* that link neighboring elements within the same string and *bridges* that connect (map) between different strings. Thus, a possible solution to the problem presented above as an example can be that ***xyz*** is transformed into ***wyz***. This solution might rely, among other things, on bridging the letters *c* and *d* (across the two base strings) and a corresponding bridge between *x* of the given target string and *w* of the constructed solution string.

The program considers rather elementary information regarding the nature and characteristics of the alphabet letters. *a* is recognized as the first letter, and *z* is recognized as the last one. The alphabetic order, i.e. the identities of the letters that come before and after each letter (successor and predecessor relations), is known as well. *Slipnet* is the name of a central ingredient in the Copycat architecture, which stores and process this information in addition to other information that is gathered during the run. It consists of approximately 60 nodes. The values associated with the Slipnet nodes form a vector, referring globally, across the whole system, to the intensity of both basic features and relations as the ones mentioned (*first letter*, *last letter*, *successor*, *predecessor*, and also *the letter A*, *the letter B*, …, *the letter Z*) and, additionally, features and relations that are not specified in advance but rather emerge during Copycat's execution. Examples for these emerging attributes (*meta-features*) are the notion of *LETTER*, *SUBSEQUENCE LENGTH*, *INCREASING SEQUENCE* (and *DECREASING SEQUENCE*) and *OPPOSITE*.

In order to concretize further the framework described above, we draw the following simplistic partial example. Suppose that one of the base strings and the given target string are ***aab*** and ***bba*** and ignore for the moment the other details of the Copycat setting (which are not much relevant in terms of our current work). Suppose further that the following grouping pattern has been evolved: in ***aab*** the first two letters are grouped together so that the whole string is now perceived as concatenation of ***aa*** and ***b***. Similarly, ***bba*** is perceived as a ***bb*** group concatenated with ***a***. This grouping pattern is not guaranteed to emerge in a real run, but seems probable in view of actual test cases described by Hofstadter et al. The ***aa*** and ***bb*** groups are likely to be marked by the features *letter A* and *letter B*, respectively, as well as by the feature *length 2*. The solution where ***aa*** is bridged to ***bb*** and ***b*** is bridged to ***a*** would have an increasing impact on the Slipnet value associated with the feature *SUBSEQUENCE LENGTH*, reflecting that the matched groups share the same length. An alternative conceivable solution, where ***aa*** is matched with ***a*** while ***b*** is matched with ***bb*** would result in increase in the value associated with the features *LETTER*, reflecting that the matched groups consist of the same letter, and *OPPOSITE*, reflecting that the groups are matched in their reverse order.

To summarize, the computational machinery underlying Copycat is based on three main factors that constantly change, while interdependently affecting each other. These are the Slipnet global values, the current setting of letter grouping, bonds and bridges, and the purposed solution string, which is firstly constructed after some execution time but from this point on, is also constantly adapted to fit the current state of the other factors and at the same time affects them.

2.2.2.2 Further Discussion in View of Methods Reviewed Previously

The computational framework of Copycat is somewhat reminiscent of the stochastic data clustering methods (Subsection 2.1.4.5), and of the scheme of constantly adjusted assignments and gradually stabilizing probabilistic or deterministic centroids (2.1.4.6 and 2.1.4.4). Update steps – grouping and ungrouping, setting and unsetting of bonds and bridges, attaching labels (features) to groups, adapting Slipnet values, and so on – are stochastically chosen from a pool of prioritized *codeletes*: small code segment that are randomly chosen to be performed. In difference from some of the methods we described, choosing the next step, i.e. codelete, to perform is not based on a global one-valued cost criterion but rather on a variety of global and local considerations (that are assessed by some codeletes dedicated for this purpose).

Among the various considerations being constantly assessed, there is a global computational-temperature parameter, which is described as regulating a global level of “open-mindedness” expressed through stochasticity level and overall likelihood of certain types of codeletes to perform. In relation to that, the temperature quantifies a global “confidence” level with regard to the currently posed solution, so that once it goes below a certain level the probability of terminating the run increases. The temperature has no deterministic cooling schedule as in simulated annealing (2.1.4.5) or in the IB method (described in Chapter 5).

Evolving representation, which is not hand coded or pre-determined, seems to be a unique and fascinating aspect of the Copycat project. On the other hand, Copycat employs a complex and hard-to-analyze computational mechanism and at the same time it manifestly gives up addressing practical real-world problems. The data-clustering-based methods we introduce later in this work (Chapter 4 and Chapter 5) attempt to maintain the flavor of creative gradually-emerging analogy discovery, along with fairly tractable computational rational and mechanisms and implementation to real-world data.

Chapter 3: Setting and Evaluation

In this chapter, we start laying the grounds to our conception of how to adapt the data-clustering framework, reviewed in the first part of the previous chapter, to the problem of drawing analogies between distinct systems – a problem that is illustratively discussed in the second part of the previous chapter. This chapter describes the basic setting. It explains how the systems to be compared, which are not necessarily similar to one another, are represented within our extended framework and what sorts of clusters are interpretable as conveying analogies or correspondences between the analogized systems. The chapter continues with a preliminary example of an application to real-world textual data of the type treated in depth in the next chapters.

In the last part of this chapter we describe how, given a configuration of clusters of the appropriate kind, the quality of the analogy or the correspondence being drawn is to be evaluated. The evaluation methods are essentially the same ones used to evaluate standard data clustering, but our extended framework suggests some subtle distinctions from the standard framework.

3.1 Problem Setting

The problem examined in this work extends the standard single-set data-clustering problem. In distinction from the setting in the single-set problem, the data for the extended problem is pre-divided into several distinct subsets of elements to be clustered. A setting of two subsets is studied first (Chapter 4). More general setting, which allows a larger number of subsets, is examined later (Chapter 5). Each one of the subsets represents one of two or more systems between which we draw an analogy or a correspondence.

A correspondence between the given subsets is established by means of partitioning them to corresponding partitions. We term each one of the subset parts that result from these partitions a *sub-cluster*. Every one of the obtained sub-clusters has a matching sub-cluster in the other subset (or several matches, one in each subset, in case the data is pre-divided to more than two subsets). Hence, a one-to-one map is established between the sub-clusters of one subset and those of the other subsets. In a setting restricted to two pre-given subsets, a pair of matched sub-clusters is termed a *coupled cluster*. A configuration of an element set pre-divided to two subsets and partitioned into three coupled clusters is sketched in Figure 3.1. Later, when a larger number of pre-given subsets is allowed, a more general term, *cross-partition cluster* will be employed to denote a collection of matched sub-clusters, one from each of the subsets. As a rule, we use the more general latter term, unless the setting under discussion is clearly of the type restricted to two subsets.

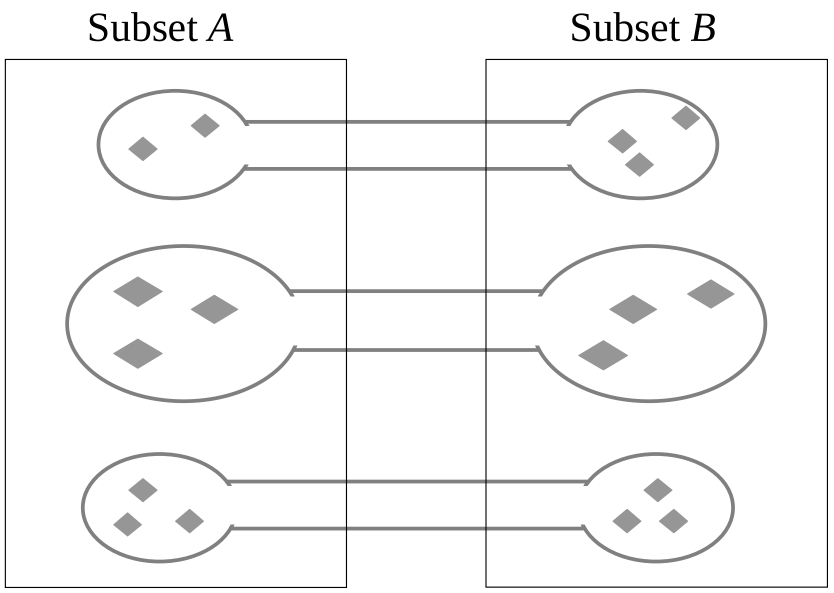


Figure 3.1: An example of a coupled-clustering configuration. The diamonds represent elements of the two pre-given subsets *A* and *B*. Closed contours represent coupled clusters, each of which links two corresponding sub-clusters each from a different subset.

Similarly to the single-set clustering problem discussed in the previous chapter (and many other problems likewise), obtaining a good solution to the cross-partition clustering task, or determining whether a given solution is satisfactory or not, is a matter of optimization over an array of potentially contradicting biases. Standard clustering aims at homogeneous subsets: each cluster is expected to consist of elements that are similar to one another (as much as possible) and, at the same time, its elements are expected to be not similar to elements in other clusters. In the case of coupled and cross-partition clustering, a new requirement is added. On one hand, we still aim at getting homogenous groups of elements. On the other hand, we want to ignore the impact of specificities characterizing any particular pre-given subset. Rather, we require that each homogenous group of elements extracted from one of the subsets would also have a good match – a corresponding group of similar elements – in the other subset or subsets, so that a cross-partition cluster is formed. An optimal configuration would thus consist of clusters containing elements that are similar to one another and distinct from elements in other clusters, subject to the context imposed by the requirement to match sub-clusters from the different subsets.

Similarly to standard-clustering formulations, the computational methods addressing the cross-partition clustering task – including the ones developed in the next chapters – might encounter various sorts of difficulties that are related to the ill-posed nature of the problem (see previous chapter, Subsection 2.1.1.2). In this respect, the situation is not improved in the cross-partition case, as this task introduces yet another source of potential biases that might not be satisfied in full, additionally to the biases already present in the original data clustering problem. The combination of considerations and biases discussed above defines cross partition clustering as a new task that cannot be tackled through previously studied methods and, particularly, not by standard data clustering methods as explained below.

Technically, it is straightforward to generate a cross-partition clustering configuration by simply ignoring the given pre-partition to subsets and applying a standard clustering method to their union. The current work focuses on cases where a standard single-set clustering method *cannot* work well. Such cases are characterized by relatively homogenous subsets, where the similarity between elements from the same subset is overall higher than similarity between elements originating in different sets. Standard clustering is committed solely to producing homogenous clusters. Therefore, a standard clustering method might tend to produce clusters that coincide with the pre-given subsets or, in case larger number of clusters is requested, clusters that are restricted to elements of an individual subset.

As mentioned in the previous chapter, the case where the pre-given subsets are relatively homogenous but, overall, are not very similar to one another is interesting, as it is characteristic of analogy making. Particularly in such cases, a solution consisting of clusters that are exclusive to one of the pre-given subsets would not reveal correspondence between the subsets. To prevent this non-favorable type of solution to the cross partition clustering problem, our method will be required to include representatives from all subsets in every cluster, along with the basic direction of including similar elements in a cluster. This additional requirement to create clusters that cut across the pre-given partition, while neutralizing regularities internal to specific subsets, differentiates the cross partition clustering problem introduced in this work from the standard data clustering problem.

3.2 A Real-world Example

In principle, cross-partition clustering seems to be applicable to revealing corresponding element groups across any unstructured set of elements pre-divided to several subsets, regardless of the type of data. Hypothetical applicative uses might include: aligning corresponding collections of atomic image components, such as pixels or contours, in order to identify corresponding objects; revealing matches between sets of physiological or psychological records in order to identify equivalencies across distinct subjects or populations and so on.

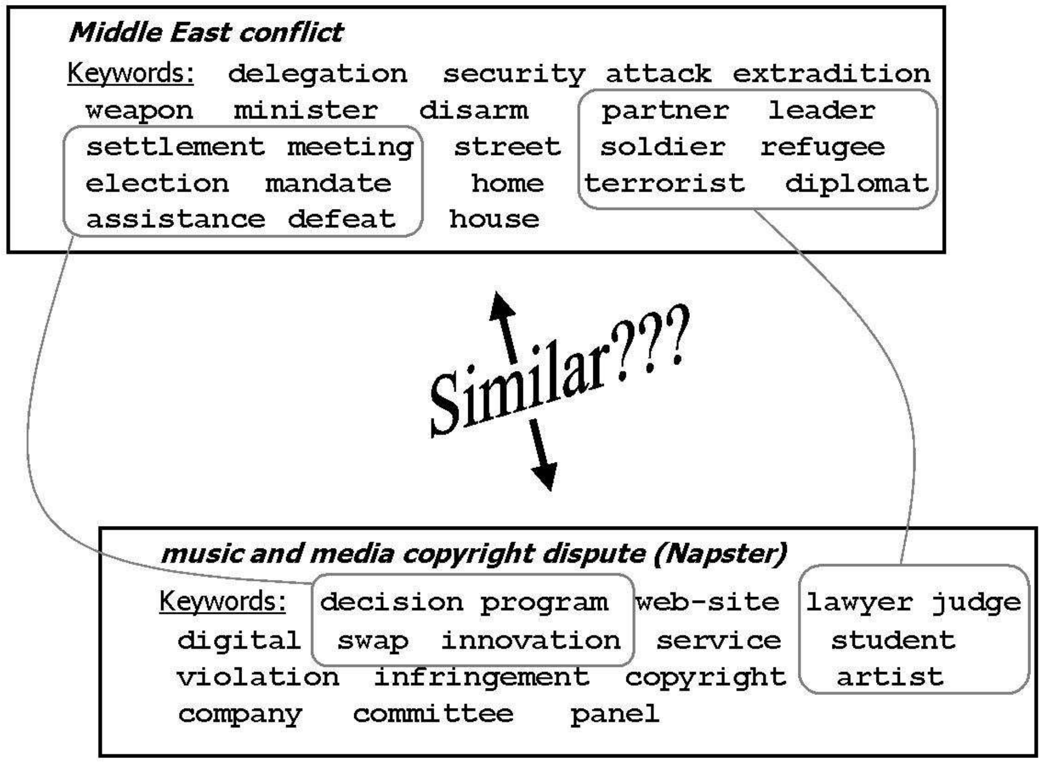


Figure 3.2: Keyword samples from news articles regarding two conflicts. Examples of coupled clusters, each consisting of two matched topical sub-clusters, are marked by curved contours.

The current work applies cross-partition clustering to another task: identification of corresponding topics across texts. Specifically, we have applied the coupled clustering and the cross partition clustering methods to collections of documents containing information regarding distinct domains. The target is to identify prominent sub-topics, themes and categories for which a correspondence can be drawn across the domains. Each domain is characterized by its own terminology and key-concepts extracted from an appropriate corpus. The keyword sets in Figure 3.2, for instance, have been extracted from news articles regarding two conflicts of distinct types: the Middle-East conflict and the dispute over copyright of music and other media types (the “Napster case”). The question of whether and with relation to which aspects these two conflicts are similar does not seem amenable to an obvious straightforward analysis. Figure 3.2 demonstrates non-trivial correspondences that have been identified by our method. For example: the role played within the Middle East conflict by individuals such as ‘soldier’, ‘refugee’, ‘diplomat’ has been aligned by our procedure, in this specific comparison, with the role of other individuals: ‘lawyer’, ‘student’ and ‘artist’ in the copyright dispute.

3.3 Evaluation

Given the ill-posed nature of the standard data-clustering problem, quantitative assessment of the performance of clustering methods is known to be a subtle issue. As the cross partition task is inherently more complex than the standard singe set task, it is at least as problematic to evaluate. The output configuration is expected to balance different types of potentially opposing biases, as discussed in the first section of this chapter, so that the judged quality of the balance obtained might be subjective or application-dependent, just as the case is with solutions to the basic clustering problem.

In general, there are two main strategies for evaluating the quality of a clustering configuration: *internal* criteria and *external* measures. An internal criterion relies solely on the processed data. Using an internal criterion can be an obvious choice when there is a known objective function exactly articulating, in a definite unquestionable manner, what is expected from the clustering mechanism. For many if not most applications, this prerequisite is not met. Our experience with cost functions (see Chapters 4 and 5) particularly demonstrates that cost scores do not capture in general the relative quality of cross-partition clustering configurations with different numbers of clusters. Likewise, relatively small cost differences of configurations resulting from a large similarity matrix often do not reflect differences in the applicative utility of the assessed configurations, even in comparing configurations of equal number of clusters. And, as implied by the ill-posed nature of the task, there are in general several alternative cost functions that may apply to the same data and there is no general procedure to determine which one is the "right" one (also, if we knew the ultimate cost function, we would direct our method to optimize it). Therefore, in this work evaluation of the results is carried out through external measures.

An external evaluation criterion assesses the results relatively to some external "gold standard" – a given configuration $\boldsymbol{E}$ from an authoritative source, assumed to provide the correct solution to the problem. We term the "gold clusters" $e_1, e_2, \ldots, e_l$ that form the criterion configuration $\boldsymbol{E}$, *classes*, to distinguish them from the automatically generated clusters $c_1, c_2, \ldots, c_k$, forming the configuration $\boldsymbol{C}$ that we wish to evaluate.

In our experiments with synthetic data (Sections 4.3 and 5.4.1), the gold standard is given by construction: each data element is drawn as part of some pre-defined class. To evaluate these experiments, we formulate in the following subsection a rather simple and straightforward external criterion, named *purity*.

In many real-world clustering problems, for example topical clustering of keywords, it is not as clear-cut in advance where each element should belong. In such cases (including the present work, see Subsection 4.4.2.3), human judges are often requested to produce the criterion set for evaluation, based on their judgment and knowledge. Depending on factors such as the specific data type and content and the level of expertise of the human judge, classes produced by human judges might turn to be just a rough criterion for evaluation (so the term “gold standard” might be a bit misleading). In the lack of precise criterion, we collected subjective criterion classes from several participants, so that the level and scope of agreement between them can be inspected as well. We have evaluated those more subtle cases through *Jaccard coefficient* described in Subsection 3.3.2 (several additional external evaluation methods are reviewed and analyzed by Meila, 2003).

3.3.1 Cluster Purity

One straightforward method for measuring the quality of a clustering configuration $\boldsymbol{C}$ in comparison to an external criterion $\boldsymbol{E}$ is known as cluster *purity*. Purity considers all elements of a cluster c in $\boldsymbol{C}$ as if they are classified as members of c's *dominant* class, which is the class e in $\boldsymbol{E}$ with which c shares maximal number of elements. For an individual cluster c, purity is defined as the ratio between those elements shared by c and e, to the total number of elements in c:

$$PURITY_{\boldsymbol{E}}(c) = \frac{1}{|c|} max_{e \in \boldsymbol{E}} \{|c \cap e|\}, \tag{3.1}$$

where $|c \cap e|$ is the number of elements shared by c and e, and $|c|$ is the total number of elements in c. Note that some classes may not share maximal number of elements with any cluster and, complimentarily, several different clusters may share the maximal intersection with the same class.

To evaluate the entire clustering configuration $\boldsymbol{C}$, given the class configuration $\boldsymbol{E}$, compute the average of the cluster-wise purities weighted by the cluster size, which sums up to:

$$PURITY_{\boldsymbol{E}}(\boldsymbol{C}) = \frac{1}{N} \sum_{c \in \boldsymbol{C}} max_{e \in \boldsymbol{E}} \{|c \cap e|\}, \tag{3.2}$$

where N is the total number of data elements.

Purity is a reliable evaluation measure under certain conditions. We use it wherever the criterion is definite and the target number of clusters is known. When it is known that the classes of $\boldsymbol{E}$ provide just a rough approximation of the desired outcome rather than a definite solution, there are several subtleties to consider and more appropriate methods. For instance, incrementing the number of clusters would tend to improve purity, up to the perfect purity of the non-informative partition to singletons. Hence, if the criterion at hand is only an approximation, one might prefer not to restrict the produced output to configurations with number of clusters identical to the number of classes in $\boldsymbol{E}$, neither to commit to any other fixed number of clusters. As these considerations are relevant to our actual experiments and, particularly, we study problems where the number of clusters is not known in advance, we evaluate our results with *Jaccard coefficient*.

3.3.2 Jaccard coefficient

Jaccard coefficient is one of several methods based on element-pair counting (used for evaluating data clustering results also by Ben-Dor, Shamir, & Yakhini, 1999). It symmetrically captures the agreement between an evaluated clustering configuration $\boldsymbol{C}$ and an external classification $\boldsymbol{E}$, on assigning pairs of data elements to the same cluster versus different clusters. A noticeable advantage of the Jaccard coefficient on other pair count method is that it does *not* incorporate those pairs about which the evaluation criterion and evaluated configuration agree that they should not be included in the same cluster. As Ben-Dor et al. (1999) note, this type of agreement is overstressed as the number of clusters grows. Meila, 2003 suggests an alternative: a set-intersection based criterion with information-theoretic motivation that claim to accommodate well to configurations of varied number of clusters. However, it is not clear whether this suggestion is straightforwardly applicable in our case: the fact that our method is applied to a pre-divided element set and the obtained clusters, which are composed of several sub-clusters, might affect the results in a manner that is not trivial to quantify. Therefore, we stick to the pair-count based Jaccard measure, for which incorporating the cross-partition aspect is simpler.

We first introduce the Jaccard measure for the standard deterministic ("hard") clustering case, where each element is assigned to one, and only one, cluster and one criterion class.

The following 0/1 valued functions, are defined for every pair of data elements x and x':

$$\begin{aligned} &\textit{Co-assign}_{\boldsymbol{C}}(x, x') \;=\; 1 \text{ iff there is } c\in\boldsymbol{C} \text{ such that } x,x'\in c \text{ (0 otherwise);}\\ &\textit{Co-assign}_{\boldsymbol{E}}(x, x') \;=\; 1 \text{ iff there is } e\in\boldsymbol{E} \text{ such that } x,x'\in e \text{ (0 otherwise).}\end{aligned} \tag{3.3}$$

Now, we define pair counts on which the Jaccard coefficient is based.

$$a_{11} = \textstyle\sum_{x,\, x'\in \boldsymbol{X}} \min\{\ \textit{Co-assign}_{\boldsymbol{C}}(x, x')\,,\ \textit{Co-assign}_{\boldsymbol{E}}(x, x')\ \} \tag{3.4a}$$

(the number of relevant data element pairs assigned into the same cluster by both $\boldsymbol{E}$ and $\boldsymbol{C}$);

$$a_{01} = \textstyle\sum_{x,\, x'\in \boldsymbol{X}} \min\{\ 1 - \textit{Co-assign}_{\boldsymbol{C}}(x, x')\,,\ \textit{Co-assign}_{\boldsymbol{E}}(x, x')\ \} \tag{3.4b}$$

(the number of pairs that have been assigned into the same cluster by $\boldsymbol{E}$ but not by $\boldsymbol{C}$);

$$a_{10} = \textstyle\sum_{x,\, x'\in \boldsymbol{X}} \min\{\ \textit{Co-assign}_{\boldsymbol{C}}(x, x')\,,\ 1 - \textit{Co-assign}_{\boldsymbol{E}}(x, x')\ \} \tag{3.4c}$$

(the number of pairs that have been assigned into the same cluster by $\boldsymbol{C}$ but not by $\boldsymbol{E}$).

Note that Jaccard coefficient ignores a_{00}, which is the agreement between $\boldsymbol{C}$ and $\boldsymbol{E}$ on those pairs that are *not* included in the same clusters. In general a_{00} becomes non-informatively dominant as the number of classes grows.

Jaccard coefficient is defined as

$$\textit{J}\textsc{accard}_{\boldsymbol{E}}(\boldsymbol{C}) = \frac{a_{11}}{a_{11} + a_{10} + a_{01}} . \tag{3.5}$$

3.3.2.1 Probabilistic Extension for Jaccard coefficient

In the previous chapter we have mentioned data-clustering methods that produce probabilistic output: each element x is distributed over all clusters, so that the association level of x with a cluster c, denoted $p(c\,|\,x)$, satisfies $\sum_{c\in \boldsymbol{C}} p(c\,|\,x) = 1$ (see previous chapter, Subsection 2.1.2.2). In order to extend the Jaccard coefficient for probabilistic clustering, we modify the definition of the binary function *Co-assign*$_{\boldsymbol{C}}$ (Eq. 3.3). The new variation provides a probabilistic value, between 0 and 1, quantifying the level by which two elements x and x' are assigned the same way by a probabilistic $\boldsymbol{C}$:

$$\textit{Co-prob-assign}_{\boldsymbol{C}}(x, x') = \sum_{c\in \boldsymbol{C}} \min\{\, p(c\,|\,x), p(c\,|\,x')\,\} . \tag{3.6a}$$

This value is equal to 1 if and only if the distributions over the clusters conditioned on both elements are identical. It coincides with the hard clustering case, whenever $p(c\,|\,x)$ and $p(c\,|\,x')$ are both 0 or 1.

The same probabilistic setting might apply also within the classification criterion $\boldsymbol{E}$: an element may be known, or approximated, as belonging to several criterion classes with either equal or varying levels of assignment (for example, a keyword might be assigned to several sub-topical keyword classes). It is natural, in such case, to require probabilistic assignment levels $p(e\,|\,x)$, so that $\sum_{e\in \boldsymbol{E}} p(e\,|\,x) = 1$. We modify accordingly the definition of *Co-assign*$_{\boldsymbol{E}}(x, x')$:

$$\textit{Co-prob-assign}_{\boldsymbol{E}}(x, x') = \sum_{e\in \boldsymbol{E}} \min\{\, p(e\,|\,x), p(e\,|\,x')\,\} . \tag{3.6b}$$

Replacing *Co-assign*$_{\boldsymbol{C}}$ and *Co-assign*$_{\boldsymbol{E}}$ in the definitions of a_{11}, a_{10} and a_{01} (Eq. 3.4a-c) by the newly defined *Co-prob-assign*$_{\boldsymbol{C}}$ and *Co-prob-assign*$_{\boldsymbol{E}}$, we may use the same definition of $\textit{J}\textsc{accard}_{\boldsymbol{E}}(\boldsymbol{C})$ (Eq. 3.5) to define a new measure, $\textit{J}\textsc{accard-prob}_{\boldsymbol{E}}(\boldsymbol{C})$, that is usable for evaluating probabilistic clustering results.

The opportunity of evaluating probabilistic clustering brings to mind the question of what is the actual target of probabilistic clustering: does it intend to expose in detail real ambiguities that are present in the data, or is it just a strategy to approximate a deterministic clustering configuration (eventually, the one where every element x is assigned to the cluster c with highest $p(c|x)$)? Both targets are of course legitimate but exposing true ambiguities is a much bigger challenge, as this implies a much richer search space and therefore "noisier" outcome. In practice, it has indeed turned that our actual probabilistic clustering results (Subsection 5.4.2) were scored slightly lower relatively to the corresponding deterministic configurations, so we concentrated on evaluating the deterministic highest-$p(c\,|\,x)$ configurations as a replacement to the raw probabilistic outcome.

In distinction from the possible interpretations of probabilistic clustering, whenever multi-assignments are present in the criterion classes as happened occasionally in our experiments, they cannot be interpreted as related with any deterministic assignments. This is the reason that in spite of resorting to deterministic configurations, we still had to use *JACCARD-PROB* rather then the standard *JACCARD* scores. We did so both when we applied deterministic clustering method (Chapter 4) and when we evaluated the deterministic approximation of a probabilistic outcome (Chapter 5).

3.3.2.2 Adapting Jaccard coefficient for the Cross Partition Setting

We propose also a variation of the Jaccard evaluation score that is specifically adapted to the cross-partition clustering setting. In the cross-partition setting, the data is pre-divided to two or more disjoint subsets. Replace a_{11}, a_{01}, and a_{10} defined above (Eq. 3.4) by corresponding quantities, with sums that include only pairs of elements from distinct subsets:

$$\begin{aligned} a'_{11} &= \textstyle\sum_{x,\,x' \text{ from distinct subsets}} \textit{Co-assign}_{\boldsymbol{C}}(x,x') \cdot \textit{Co-assign}_{\boldsymbol{E}}(x,x'), \\ a'_{01} &= \textstyle\sum_{x,\,x' \text{ from distinct subsets}} (\,1 - \textit{Co-assign}_{\boldsymbol{C}}(x,x')\,) \cdot \textit{Co-assign}_{\boldsymbol{E}}(x,x'), \\ a'_{10} &= \textstyle\sum_{x,\,x' \text{ from distinct subsets}} \textit{Co-assign}_{\boldsymbol{C}}(x,x') \cdot (\,1 - \textit{Co-assign}_{\boldsymbol{E}}(x,x')\,). \end{aligned} \tag{3.7}$$

Plugging a'_{11}, a'_{11}, a'_{11} into the definition of Jaccard coefficient (Eq. 3.5), we obtain:

$$\textit{JACCARD-CP}_{\boldsymbol{E}}(\boldsymbol{C}) \;=\; \frac{a'_{11}}{a'_{11}+a'_{10}+a'_{01}}, \tag{3.8}$$

a variation on Jaccard coefficient adapted to the context of the new problem, which considers only elements from the distinct subsets and excludes the impact of all within-subset pairs. If we also replace in Eq. 3.7 above *Co-assign* by *Co-prob-assign* (Eq. 3.6), we obtain the *JACCARD-PROB-CP* measure, which combines probabilistic clustering with ignoring the impact of the within-subset pairs.

Similarly to the corresponding question regarding evaluation of probabilistic clustering, the question of whether to use *JACCARD-CP* as a replacement for the original *JACCARD* measure depends on how exactly one views the target of the cross-partition task. If the emphasis is on creating a variety of associations between the pre-given subsets, then it would make sense to use *JACCARD-CP* (or *JACCARD-PROB-CP*). If, in subtle distinction, the focus is on revealing concepts or themes that cut across the pre-given partition and might nevertheless incorporate some within-subset information, then using the original *JACCARD* measure (or *JACCARD-PROB*) can be viewed as more appropriate. In evaluating our actual results, we accordingly use *JACCARD-PROB-CP* for evaluating our coupled clustering method (Section 4.4) that, as will be explained, relies solely on cross-subset information. For evaluating the later cross-partition method (Subsection 5.4.2), which does not ignore within-subset information, we use the *JACCARD-PROB* measure.

Chapter 4: Coupled Clustering

The method introduced in this chapter – *coupled clustering* – extends the standard clustering task for a set of data elements pre-divided into two disjoint subsets. As explained in the previous chapter, we study the problem of partitioning in parallel the pair of pre-given subsets to groups of elements, or sub-clusters, each of which is matched with a corresponding sub-cluster in the other subset. A pair of matched sub-clusters forms together a *coupled cluster*.

This chapter relies on a paper introducing the coupled clustering framework (Marx et al., 2002), extending earlier publications (Marx & Dagan, 2001; Marx, Dagan & Buhmann, 2001). In Section 4.1, we review the computational methods that underlie our approach, namely the standard clustering method by Puzicha, Hofmann & Buhmann (2000) and co-occurrence based similarity measures by Lin (1998) and Dagan, Marcus & Markovitch (1995). The coupled clustering method is formally introduced in Section 4.2. Then, we demonstrate our method on synthetic data (Section 4.3) and on a task of detecting equivalencies across distinct textual corpora (Section 4.4). The examined corpora deal with the conflict theme as exemplified in Figure 3.2 and on various religions between which we identify correspondences. The religion data is thoroughly evaluated through comparison of our program's output with keyword classes that were formed manually by experts of comparative studies of religions. Section 4.5 concludes this chapter with further discussion.

4.1 Computational Background

This section reviews the two computational frameworks that form the basis for the coupled-clustering method. The first subsection concentrates on the relevant details of a data-clustering method, by Puzicha, Hofmann & Buhmann (2000), which our algorithm extends for coupled clustering. The next subsection reviews methods for calculating the similarity values used as input for our method.

4.1.1 Cost-based Pairwise Clustering

Puzicha, Hofmann & Buhmann (2000) present, analyze and classify a family of pairwise clustering cost functions. Their framework assumes "hard" assignments: every data element is assigned into one and only one of the clusters. In reviewing their work we use the following notation. A data clustering procedure partitions the elements of a given dataset, $\boldsymbol{X}$, into disjoint element clusters, $c_1, \ldots, c_k$. The number of clusters, k, is pre-determined and specified as an input parameter to the clustering algorithm. A cost criterion guides the search for a suitable clustering configuration. This criterion is realized through a cost function $H(S, \boldsymbol{C})$ taking the following parameters:

(i) $S = \{s_{xx'}\}_{x,x' \in \boldsymbol{X}}$: a collection of pairwise similarity values[1], each of which pertains to a pair of data elements x and x' in $\boldsymbol{X}$.

(ii) $\boldsymbol{C} = (c_1, \ldots, c_k)$: a candidate clustering configuration, specifying assignments of all elements into the disjoint clusters (that is $\bigcup c_j = \boldsymbol{X}$ and $c_j \cap c_{j'} = \phi$ for every $1 \le j \le j' \le k$).

The cost function outputs a numeric cost value for the input clustering-configuration $\boldsymbol{C}$, given the similarity collection S. Thus, various candidate configurations can be compared and the best one, i.e the configuration of lowest cost, is chosen. The main idea underlying clustering criteria is the preference of configurations in which similarity of elements within each cluster is generally high and similarity of elements that are not in the same cluster is correspondingly low. This idea is formalized by Puzicha et al. through the *monotonicity* axiom: in a given clustering configuration, increasing similarity values, pertaining to elements within the same cluster, cannot increase the cost assigned to that configuration. Similarly, increasing the similarity level of elements belonging to distinct clusters cannot improve the cost.

Monotonicity captures the most basic intuitive expectation from pairwise data clustering. By introducing further requirements, Puzicha et al. focus on a more confined family of cost functions. The following requirement focuses attention on functions of relatively simple structure. A cost function H fulfills the *additivity* axiom if it can be presented as the cumulative sum of repeated applications of "local" functions referring individually to each pair of data elements. That is:

$$H(S, \boldsymbol{C}) \;=\; \sum_{x,x' \in \boldsymbol{X}} \psi^{xx'}(x, x', s_{xx'}, \boldsymbol{C}) \,, \qquad (4.1)$$

where $\psi^{xx'}$ depends on the two data elements x and x', their similarity value, $s_{xx'}$, and the whole clustering configuration $\boldsymbol{C}$. An additional axiom, the *permutation invariance* axiom, states that cost should be independent of element and cluster reordering. Combined with the additivity axiom, it implies that a single local function ψ, s.t. $\psi^{xx'} \equiv \psi$ for all $x,x' \in \boldsymbol{X}$, can be assumed.

Two additional invariance requirements aim at stabilizing the cost under simple transformations of the data. First, relative ranking of all clustering configurations should persist under scalar multiplication of the whole similarity ensemble. Assume that all similarity values within a given collection S are multiplied by a positive constant η, and denote the modified collection by ηS. Then, H fulfills the *scale invariance* axiom if for every fixed clustering configuration $\boldsymbol{C}$, the following holds:

[1] In their original formulation, Puzicha et al. use distance values (dissimilarities) rather then similarities. Hereinafter, we apply straightforward adaptation to similarity values by adding a minus sign to H. Adhering to the cost minimization principle, this transformation replaces the cost paid for within-cluster dissimilarities with cost saved for within-cluster similarities (alternatively pronounced as "negative cost paid").

$$H(\eta S, \boldsymbol{C}) \;=\; \eta H(S, \boldsymbol{C})\,. \tag{4.2}$$

Likewise, it is desirable to control the effect of an addition of a constant. Assume that a fixed constant Δ is added to all similarity values in a given collection S, and denote the modified collection by $S^{+\Delta}$. Then, H fulfills the *shift invariance* axiom if for every fixed clustering configuration $\boldsymbol{C}$, the following holds:

$$H(S^{+\Delta}, \boldsymbol{C}) \;=\; H(S, \boldsymbol{C}) + \Phi\,, \tag{4.3}$$

where Φ may depend on Δ and on any aspect of the clustered data (typically the data size), but not on the particular configuration $\boldsymbol{C}$.

As the most consequential criterion, to assure that a given cost function is not subject to local slips, Puzicha et al. suggest a criterion for *robustness*. This criterion ensures that whenever the data is large enough, bounded changes in the similarity values regarding one specific element, $x \in \boldsymbol{X}$, would result in limited effect on the cost. Consequently, the cost assigned to any clustering configuration would not be sensitive to a small number of fluctuations in the similarity data. Formally, denote the size of the set of elements $\boldsymbol{X}$ by N and let $S^{x+\Delta}$ be the collection obtained by adding Δ to all similarity values in S pertaining to one particular element, $x \in \boldsymbol{X}$. Then H is robust (in the strong sense) if it fulfills

$$\tfrac{1}{N}\,|\,H(S, \boldsymbol{C}) - H(S^{x+\Delta}, \boldsymbol{C})\,| \xrightarrow[N\to\infty]{} 0. \tag{4.4}$$

Puzicha et al. show that any cost function satisfying Equations 4.1, 4.2, 4.3 is a linear combination of two factors: a positive component (to be minimized) incorporating averages of distances between elements within the same cluster, and a negative component (to be maximized) incorporating averages of distances between elements from different clusters. It turns out that among those cost functions there is only one function that satisfies the strong robustness criterion of Equation 4.4 in addition to Equations 4.1, 4.2, 4.3. This function, denoted here as H^0, involves only similarity values pertaining to elements within the same cluster (*within-cluster similarities*).

Specifically, H^0 is a weighted sum of average within-cluster similarity. Denote the sizes of the k clusters $c_1, \ldots, c_k$ by $n_1, \ldots, n_k$ respectively. The average within-cluster similarity for the cluster c_j is then

$$Avg_j \;=\; \frac{\sum_{x,x'\in c_j} s_{xx'}}{n_j \times (n_j - 1)}\,. \tag{4.5}$$

H^0 weights the contribution of each cluster to the cost proportionally to the cluster size:

$$H^0 \;=\; -\textstyle\sum_j n_j Avg_j\,. \tag{4.6}$$

In Section 4.3, we modify H^0 to adapt it for the coupled clustering setting.

4.1.2 Feature-based Similarity Measures

In our calculations, similarity between data elements is assessed through methods that take feature vectors as input and put heavier weights on the more informative features. The information regarding a data element, *x*, conveyed through a given feature, *y*, is assessed through the following term:[2]

$$I(x,y) \;=\; \log_2^+ \frac{p(x \mid y)}{p(x)} \;, \qquad (4.7)$$

where, *p* denotes conditional and unconditional occurrence probabilities and the '+' sign indicates that 0 is returned whenever the $\log_2$ function produces negative value. In our experiments *x* and *y* are generally words and *p* is empirical occurrence probability.

Dagan, Marcus & Markovitch (1995) base their similarity measure on the following term:

$$sim_{DMM}(x,x') \;=\; \frac{\sum_y \min\{I(x,y), I(x',y)\}}{\sum_y \max\{I(x,y), I(x',y)\}} \;. \qquad (4.8)$$

The similarity value obtained by this measure is higher as the number of highly informative features, providing comparable amount of information for both elements *x* and *x*', is larger.

Lin, 1998 incorporates the information term of Equation 4.7, as well, though differently:

$$sim_L(x,x') \;=\; \frac{\sum_{\{y/I(x,)>0 \wedge I(x',y)>0\}} (I(x,y) + I(x',y))}{\sum_y (I(x,y) + I(x',y))} \;. \qquad (4.9)$$

Here, the obtained similarity value is higher as the number of features that are somewhat informative for both elements, x_1 and x_2, is larger, and the relative contribution of those is in proportion to the total information all features convey.

Similarly to the cosine measure (see 2.1.3.3), both sim_{DMM} and sim_L measures satisfy: (i) the maximal similarity value, 1, is obtained for element pairs with relation to which every feature is equally informative (including self similarity); and (ii) the minimal similarity value, 0, is obtained whenever every attribute is not informative for either one of the elements. Accordingly, our formulation and the experiments below follow the convention that a zero value denotes no similarity.

In the coupled clustering experiments on textual data that are described later, we use both above similarity measures. We utilize pre-calculated sim_L values for one experiment (Subsection 4.4.1) and we calculate sim_{DMM} values, based on word co-occurrence within our corpora, for another experiment (Subsection 4.4.2).

[2] The expectation over the term of Equation 4.7 over co-occurrences of all *x*'s and *y*'s, (with $\log_2$) is the *mutual information* of *x* and *y* (Cover and Thomas, 1991, p. 18).

4.2 Algorithmic Framework for Coupled Clustering

In this section, we define the coupled clustering task and introduce an appropriate setting for accomplishing it. We then present alternative cost criteria that can be applied within this setting and describe the search method that we use to identify coupled-clustering configurations of low cost. As we noted in Chapter 3, coupled clustering is the problem of partitioning two data subsets into corresponding sub-clusters, so that every sub-cluster is matched with a counterpart in the other subset. Each pair of matched sub-clusters forms jointly a coupled cluster. As in the standard single set task of data clustering, each coupled cluster consists of elements that are similar to one another and distinct from elements in other clusters. However, this is subject to an additional bias imposed by the requirement to match sub-clusters of each pre-given subsets with those of the other subset.

4.2.1 Directing Clustering through Between-subset Similarities

Coupled clustering divides the two pre-given element subsets, X^1 and X^2, into disjoint sub-clusters $c_1^1, \ldots, c_k^1$ and $c_1^2, \ldots, c_k^2$. Each of these sub-clusters is coupled with a corresponding sub-cluster of the other subset, that is c_j^1 is coupled with c_j^2 for $j = 1\ldots k$. Every pair of coupled sub-clusters forms a unified coupled-cluster, $c_j = c_j^1 \cup c_j^2$, which contains elements of both pre-given subsets (see Figure 4.1). We approach the coupled clustering problem through a pairwise-similarity-based setting, incorporating the elements of both X^1 and X^2. Our treatment is independent of the method by which similarity values are calculated: feature-based calculations such as those described in Subsection 4.1.2, subjective assessments, or any other method.

The notable feature distinguishing our method from standard pairwise clustering, is the set of similarity values, S, that are considered. A standard pairwise clustering procedure potentially considers the similarity values referring to all pairs of elements within the undivided clustered set. Typically, the only similarity values that are not considered are self-similarities. In the coupled clustering setting, there are two different types of available similarity values. Values of one type denote within-subset similarities (short gray arrows in Figure 4.1). Values of the second type denote similarities of element pairs consisting of one element from each subset (*between-subset similarities*; long black arrows in Figure 4.1). As an initial strategy, to be complied with throughout this chapter, we choose to ignore similarities of the first type altogether and to concentrate solely on between-subset similarities: $S = \{s_{xx'}\}$, where $x \in X^1$ and $x' \in X^2$. Consequently, the assignment of a given data element into a coupled cluster is directly influenced by the most similar elements of the other subset, regardless of its similarity to members of its own subset.

The policy of excluding within-subset similarities captures, according to our conception, the unique context posed by aligning two pre-given subsets representing distinct domains with respect to one another. Correspondences special to the current comparison, which underlie presumed parallel or analogous structure of the compared systems, are thus likely to be identified abstracted from the distinctive information characterizing each individual system. This fits our key goal of detecting commonalities, while masking out subset-internal structures (see Section 3.1). In this chapter, we do not deal with the questions of whether and how available information regarding within-subset similarities should be incorporated. The next chapter introduces a method that processes the data more comprehensively.

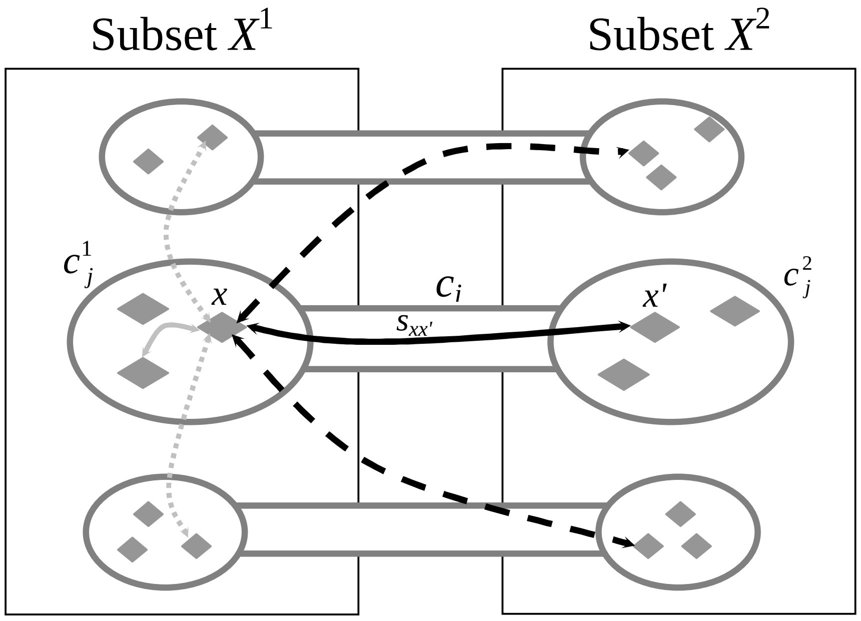


Figure 4.1: The coupled clustering setting. The diamonds represent elements of the pre-given subsets X^1 and X^2. The long black arrows represent the values in use: similarity values pertaining to two elements, one from each subset. The shorter grey arrows stand for the disregarded similarity values within a subset.

4.2.2 Three Alternative Coupled Clustering Cost Functions

Given the setting described above, in order to identify configurations that accomplish the coupled clustering task, our next step is defining a cost function. In formulating it, we closely follow the standard pairwise-clustering framework presented by Puzicha, Hofmann & Buhmann, (2000, see Subection 4.1.1 above). Given a collection of similarity values S pertaining to the members of two pre-given subsets, X^1 and X^2, we formulate an additive cost function, $H(S, \boldsymbol{C})$, which assigns a cost value to any coupled-clustering configuration $\boldsymbol{C}$. Given such a cost function and a search strategy (see 4.2.4 below) our procedure would be able to output a coupled clustering configuration specifying assignments of the elements into a pre-determined number, *k*, of coupled clusters. We concentrate on Puzicha et al.'s H^0 cost function (Subection 4.1.1, Eq. 4.6), which is limited to similarity values within each cluster and weighs each cluster's contribution proportionally to its size. Below we present and analyze three alternative cost-functions derived from H^0.

As in clustering in general, the coupled clustering cost function should assign similar elements into the same cluster and dissimilar elements into distinct clusters (as articulated by the monotonicity axiom in Subsection 4.1.1). A coupled-clustering cost function is thus expected to assign low cost to configurations in which the similarity values, $s_{xx'}$, of elements x and x' of coupled subs-clusters, c_j^1 and $c_{j'}^2$, are high on average. (The dual requirement to assign low cost whenever similarity values of elements x and x' from non-coupled sub-clusters c_j^1 and $c_{j'}^2$, $j \neq j'$, are low, is implicitly fulfilled). In addition, we seek to avoid influence of transient or minute components – those that could have been evolved from casual noise or during the optimization process – and maintain the influence of stable larger components. Consequently, the contribution of large coupled clusters to the cost is greater than the contribution of small ones with the same average similarity. This direction is realized in H^0 through weighting each cluster's contribution by its size.

In the coupled-clustering case, one apparent option is to apply straightforwardly the original H^0 cost function to our restricted collection of between-subset similarity values. The average similarity of the coupled cluster $c_j = c_j^1 \cup c_j^2$ is then calculated as

$$Avg'_j \;=\; \frac{\sum_{x\in c_j^1, x'\in c_j^2} s_{xx'}}{n_j \times (n_j - 1)}\;, \tag{4.10}$$

where n_j is the number of elements in c_j (so that Eq. 4.10 differs from the standard average formula, Eq. 4.5, by setting all within-subset similarities to 0). As in H^0 (Eq. 4.6), the average similarity of each cluster is multiplied by the cluster size. Thus, the following cost function, H^1, is obtained:

$$H^1 \;=\; -\sum_j n_j \times Avg'_j\;. \tag{4.11}$$

Alternatively, as we restrict the collection of similarities being considered in our calculations, we might want to take it into account in the averaging scheme as well. The actual number of considered similarities in the restricted collection is, for each j, the product $n_j^1 \times n_j^2$ of the sizes of the two sub-clusters c_j^1 and c_j^2 forming c_j. The following averaging scheme might seem more natural for the coupled clustering setting:

$$Avg''_j \;=\; \frac{\sum_{x\in c_j^1, x'\in c_j^2} s_{xx'}}{n_j^1 \times n_j^2}\;, \tag{4.12}$$

Correspondingly, a second cost variant, H^2, is given:

$$H^2 \;=\; -\sum_j n_j \times Avg''_j \,. \tag{4.13}$$

One factor to which the weighting schemes of H^1 and H^2 does not refer is the inner partition of the coupled clusters. Hence, we suggest yet another alternative that incorporates the proportion between the sizes of the two sub clusters, namely weighing the average similarity each cluster contributes to the cost by the geometrical mean of the corresponding coupled sub-cluster sizes: $\sqrt{n_j^1 \times n_j^2}$. This yields yet another cost function:

$$H^3 \;=\; -\sum_j \sqrt{n_j^1 \times n_j^2} \times Avg''_j \,. \tag{4.14}$$

The weighting factor of H^3 results in penalizing large gaps between the two sizes, n_j^1 and n_j^2, and in preferring balanced configurations, with coupled-cluster inner proportions maintaining the global proportion of the clustered subsets ($n_j^1 / n_j^2 \cong N^1 / N^2$ for each j, where N^1 and N^2 are the sizes of X^1 and X^2, respectively). Later on we refer to the cost function H^2 and H^3, as the "additive" and the "multiplicative" cost functions.

4.2.3 Properties of the Coupled Clustering Cost Functions

Puzicha, Hofmann & Buhmann, (2000) based their characterization of pairwise-clustering cost-functions on some properties and axioms (see Subsection 4.1.1 above). In the previous subsection, we have followed their conclusions in adapting, in three different variants, one function, H^0, that realizes the most favorable properties. It is worthwhile to see if and how these properties are preserved through the adaptation for the coupled clustering setting. As we show below, all the three cost functions that we have derived, H^1, H^2 and H^3, are additive by construction and it immediately follows that they are also scale invariant. They are not, except for H^2, shift-invariant. However, the effect of a constant added to all between-subset similarity values is bounded for H^1 and H^3, as well. Finally, H^1 and H^3 are robust (but not by H^2).

Lemma 4.1: H^1, H^2 and H^3, are additive (Eq. 4.1).

Proof: For each one of the three functions, each element pair with non-zero impact on the cost (i.e., members of the same coupled cluster from different subsets) adds to the cost a component of the form $\psi^{xx'}(x, x', s_{xx'}, \boldsymbol{C})$. This contribution amounts to the average similarity within the cluster to which both elements belong multiplied by a factor depending on this cluster. Specifically, for each pair of elements $x, x' \in c_j$, such that $x \in X^1$ and $x' \in X^2$, we have the following terms:

$$\psi_1^{xx'} = -\frac{s_{xx'}}{n_j - 1} \;,\quad \psi_2^{xx'} = -n_j \frac{s_{xx'}}{n_j^1 \times n_j^2} \,,\quad \psi_3^{xx'} = -\frac{s_{xx'}}{\sqrt{n_j^1 \times n_j^2}} \,, \tag{4.15}$$

which explicate the non-zero summands forming H^1, H^2 and H^3, respectively.

Lemma 4.2: H^1, H^2 and H^3, are scale invariant (Eq. 4.2).

Proof: As $\psi_1^{xx'}$, $\psi_2^{xx'}$ and $\psi_3^{xx'}$ of the previous lemma all depend linearly on $s_{xx'}$ (i.e., $\eta\psi_i^{xx'}(x, x', s_{xx'}, \boldsymbol{C}) = \psi_i^{xx'}(x, x', \eta s_{xx'}, \boldsymbol{C})$), it follows that the three cost functions satisfy the scale invariance property.

Lemma 4.3: H^2 is shift invariant (Eq. 4.3).

Proof: in the *j*-th cluster there are $n_j^1 \times n_j^2$ cross-subset pairs (x, x'), so that introducing a constant shift to all the considered similarities and summing $\psi_2^{xx'}$ (defined in the proof to Lemma 5.1) over all the relevant pairs within the *j*-th cluster gives:

$$\sum_{x,x' \in c_j, x \in X^1, x' \in X^2} -n_j \frac{s_{xx'} + \Delta}{n_j^1 \times n_j^2} = \left(\sum_{x,x' \in c_j, x \in X^1, x' \in X^2} -n_j \frac{s_{xx'}}{n_j^1 \times n_j^2} \right) - n_j\Delta . \qquad (4.16)$$

Taking sum over the above cluster-dependent terms of Eq. 4.16 yields $H^2(S^{+\Delta}, \boldsymbol{C}) = H^2(S, \boldsymbol{C}) + N\Delta$, where the term $N\Delta$ depends on the shift constant and on the data (and not on $\boldsymbol{C}$), as required for maintaining the shift invariance property.

Lemma 4.4: For H^1 and H^3, the effect on the cost value of adding a constant to all between-subset similarities is bounded.

Proof: Both H^1 and H^3 are are non-positive, thus bounded from above by 0.

For H^1, the *j*-th cluster's contribution to the modified cost resulting from increasing all similarities by a positive Δ is $(n_j^1 \times n_j^2 / (n_j - 1))\Delta \le \min\{n_j^1, n_j^2\}\Delta \le (n_j/2)\Delta$. Therefore, $H^1(S^{+\Delta}, \boldsymbol{C}) \ge H^1(S, \boldsymbol{C}) - (N/2)\Delta$.

Similarly, for H^3, the *j*-th cluster's contribution to the modification in cost value is $\sqrt{n_j^1 \times n_j^2}\,\Delta \le (n_j/2)\Delta$, so that also for H^3 the following holds: $H^3(S^{+\Delta}, \boldsymbol{C}) \ge H^3(S, \boldsymbol{C}) - (N/2)\Delta$.

We note that it is possible to modify H^1 and H^3 so to impose the shift-invariance property on them. For that, one can use the derivative of, say, H^3 with respect to Δ, which is the increment for all between-data-set similarity values. This is a linear function so the resulting derivative is $D = \sum_j 1 / \sqrt{n_j^1 \times n_j^2}$. Consequently, normalizing H^3 by $1/D$ would result in perfect shift invariance. However, H^3 in its non-normalized form is nearly shift-invariance with regard to configurations for which the clusters approximately maintain the global proportion of the clustered data sets X^1 and X^2, while highly imbalanced configurations are highly penalized. Since our experiments use similarity measures with values between 0 and 1, we stick to the simpler formulation of Eqs. 4.11 and 4.14 above, assuming that the normalized version would behave similarly.

Lemma 4.5: H^1 is robust (Eq. 4.4); H^3 is robust provided the ratio between the sub-cluster sizes of any coupled cluster is kept bounded as the number of elements grows.

Proof: For H^1:

$$\frac{1}{N} | H^1(S,\boldsymbol{C}) - H^1(S^{x+\Delta},\boldsymbol{C}) | \;\le\; \frac{1}{N}\max\{n_j^1,n_j^2\}\frac{\Delta}{n_j-1} \;\le\; \frac{\Delta}{N} \xrightarrow[N\to\infty]{} 0\,, \tag{4.17}$$

where j is the index of the cluster to which x is assigned. Similarly for H^3:

$$\frac{1}{N} | H^3(S,\boldsymbol{C}) - H^3(S^{x+\Delta},\boldsymbol{C}) | \tag{4.18}$$

$$\le\; \frac{1}{N}\max\{n_j^1,n_j^2\}\frac{\Delta}{\sqrt{n_j^1 \times n_j^2}} \;=\; \frac{\Delta}{N}\sqrt{\frac{\max\{n_j^1,n_j^2\}}{\min\{n_j^1,n_j^2\}}} \xrightarrow[N\to\infty]{} 0\,,$$

where convergence relies on the assumption regarding the ratio between the sub-clusters.

Finally, we note that using the coupled sizes geometrical mean as a weighting factor, H^3 tends to escape configurations that match minute sub-clusters with large ones, which are occasionally the consequence of noise in the input data or of fluctuations in the search process. It turns that this property provides H^3 with a notable advantage over H^1 and H^2, as our experiments indeed show (see Sections 4.3 and 4.4).

4.2.4 Optimization Method

In order to find the clustering configuration of minimal cost, we have implemented a stochastic search procedure, namely a variation of the simulated annealing method based on the sampling pattern of the Gibbs sampler algorithm (Geman & Geman, 1984; See also Chapter 2, Subsection 2.1.4.5). Starting with random assignments into clusters, this algorithm iterates repeatedly through all data elements and probabilistically reassigns each one of them in its turn, according to a probability governed by the expected cost change. Suppose that in a given assignment configuration, $\boldsymbol{C}$, the cost difference $\Delta_{j|x,\boldsymbol{C}}$ is obtained by reassigning a given element, x, into the j-th cluster ($\Delta_{j|x,\boldsymbol{C}} = 0$ in case x is already assigned to the j-th cluster). The target cluster, into which the reassignment is actually performed, is selected among all candidates with probability

$$p(j) \;\equiv\; p(j|x,\boldsymbol{C}) \;\prec\; \frac{1}{1+\exp\{-\beta\Delta_{j|x,C}\}} \tag{4.17}$$

Consequently, the chances of an assignment to take place are higher as the resulting reduction in cost is larger. In distinction from the original simulated annealing algorithm (Kirkpatrick, Gelatt & Vecchi, 1983), assignments that result in increased cost are possible, though with relatively low probability. The β parameter, controlling the randomness level of reassignments, functions as an inverse "computational temperature". Starting at high temperature followed by a progressive cooling schedule, that is initializing β to a small positive value and gradually increasing it (e.g. repeatedly multiply β by a constant that is slightly greater than one, 1.001 in our experiments), turns most profitable assignments increasingly probable. As the clustering process proceeds, the gradual "cooling" systematically reduces the probability that the algorithm would be trapped in a local minimum (though global minimum is fully guaranteed only under an impracticably slow cooling schedule). The algorithm execution stops after several repeated iterations through all data elements, in which no cost change has been recorded (50 iterations in our experiments).

4.3 Experiments with Synthetic Data

A set of experiments on synthetic data has been conducted for evaluating the performance of our algorithm, making use of the three cost functions introduced in Section 4.3 above. These experiments have measured, under changing noise levels, how well each of the cost functions reconstructs a configuration of pre-determined clusters of various inner proportions.

Each input similarity value (i.e. between-subset similarities) in these experiments incorporates a basic similarity level, dictated by the pre-determined clustering configuration, combined with an added random component introducing noise. The basic similarity values have been generated so that each element is assigned into one of four coupled clusters. Elements in the same cluster share the maximal basic similarity of value 1, while elements in distinct clusters share the minimal basic similarity 0. The noisy component combined with the basic value is a random number between 0 and 1.

In precise terms, the similarity value $s_{xx'}$, of any $x \in X^1$ and $x' \in X^2$ (X^1 and X^2 are the pre-given subsets), has been set to

$$s_{xx'} = (1-\alpha)\,\delta_{j(x)j(x')} + \alpha r_{xx'}, \tag{4.18}$$

where $\delta_{j(x)j(x')}$ – the basic similarity level – is 1 if $x \in X^1$ and $x' \in X^2$ are, by construction, in the same (j-th) coupled cluster or otherwise 0 and $r_{xx'}$ – the random component – is sampled uniformly between 0 and 1, differently for each x and x' in each experiment. The randomness proportion parameter α (i.e. level of added noise), also between 0 to 1, is fixed throughout each experiment, to maintain a steady average noise level.

In order to study the effect of the coupled-cluster inner proportion, we have run four sets of experiments. Given subsets X^1 and X^2 consisting of 32 elements each, four types of synthetic coupled-clustering configurations have been constructed, in which the sizes n_j^1 and n_j^2 of the sub-cluster pairs $c_j^1 \subset X^1$ and $c_j^2 \subset X^2$, together forming the j-th coupled-cluster, have been set as follows:

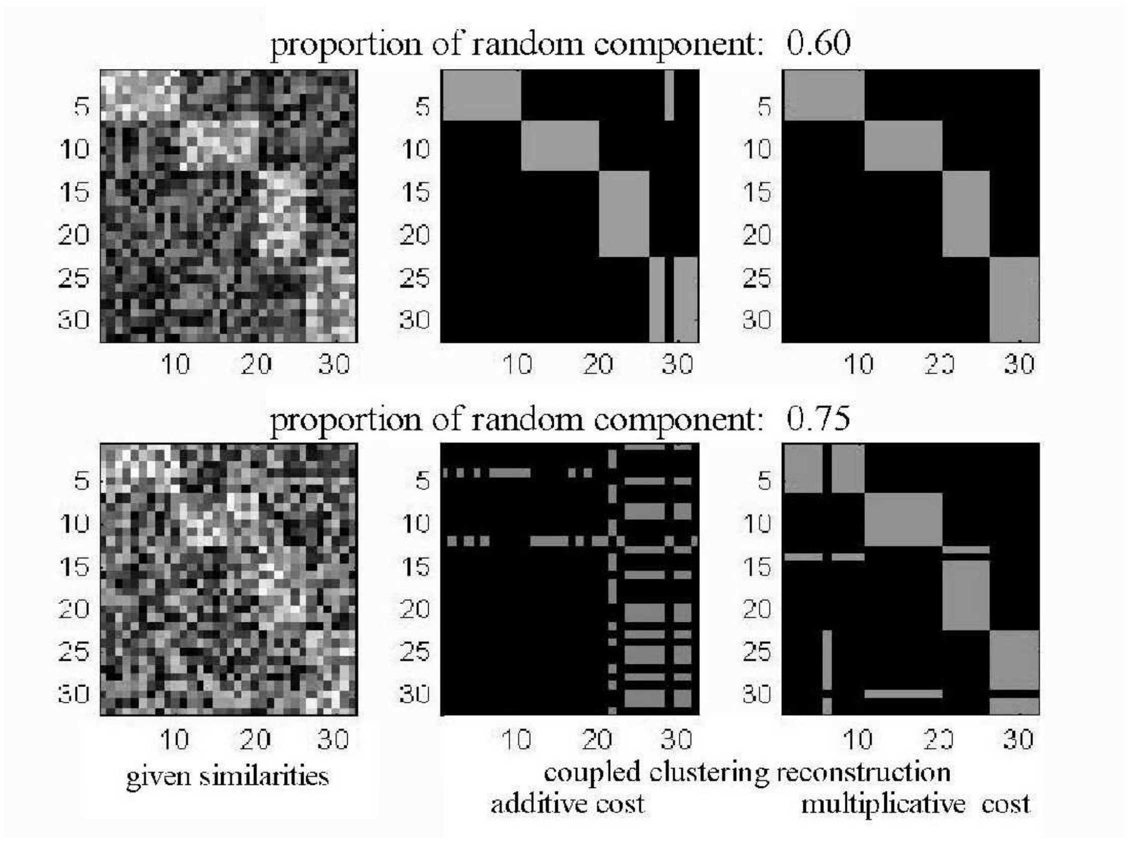


Figure 4.2: Reconstruction of synthetic coupled-clustering configurations of the '10-6 coupling' target configuration type from noisy similarity data. Lines and columns of the plotted gray-level matrices correspond to members of the two sets. On the left-hand side – original similarity values – the gray-level of each pixel represents the corresponding similarity value between 0 (black) and 1 (white). In the reconstructed data, gray level corresponds to average similarity within each reconstructed cluster. The bottom part demonstrates that the multiplicative cost function, H^3, reconstructs better under intensified noise.

(i) $n_j^1 = n_j^2 = 8$, for $j = 1\ldots4$;

(ii) $n_j^1 = 10$, $n_j^2 = 6$ for $j = 1,2$ and $n_j^1 = 6$, $n_j^2 = 10$ for $j = 3,4$;

(iii) $n_j^1 = 12$, $n_j^2 = 4$ for $j = 1,2$ and $n_j^1 = 4$, $n_j^2 = 12$ for $j = 3,4$;

(iv) $n_j^1 = 14$, $n_j^2 = 2$ for $j = 1,2$ and $n_j^1 = 2$, $n_j^2 = 14$ for $j = 3,4$.

These four configuration types, respectively labeled '8-8 coupling', '10-6 coupling', '12-4 coupling' and '14-2 coupling', have been used in the four experiment sets.

It is convenient to visualize a collection of similarity values as a gray-level matrix, where rows and columns correspond to individual elements of the two clustered subsets and each pixel represents the similarity level of the corresponding elements. The diagrams on the left-hand side in Figure 4.2 show two collections of similarity values generated with two different noise levels. White pixels represent the maximal similarity level in use, 1; black pixels represent the minimal similarity level, 0; the intermediate gray levels represent similarities in between. The middle and right-hand-side columns of Figure 4.2 displays clustering configurations as reconstructed by our algorithm, using the additive H^2 and multiplicative H^3 cost functions respectively, given the input similarity values displayed on the left-hand side. Examples from the 10-6 coupling experiment set, with two levels of noise, are displayed. Bright pixels indicate that the corresponding elements are in the same reconstructed cluster. It demonstrates that, for the 10-6 coupling, the multiplicative variant H^3 tends to tolerate noise better than the additive variant H^2 and that this advantage grows when the noise level intensifies (bottom of Figure 4.2).

The performance over all experiments in each set has been measured through the *PURITY* measure, introduced in Section 4.2. Figure 4.3 displays average accuracy for the changing noise levels, separately for each experiment set. The multiplicative cost function H^3, is biased toward balanced coupled clusters, i.e. clusters in which the inner proportion is close to the global proportion of the subsets (which are equal in size in our case). Our experiments indeed verify that H^3 reconstruct better than the other functions, particularly in cases of almost balanced inner proportions.

Figure 4.3 shows that the accuracy obtained using the restricted standard-clustering function H^1 is consistently worse than the accuracy of H^3. In addition, for all internal proportions, there is some range, on the left-hand side of each curve, in which H^3 performs better than the additive function H^2. The range where H^3 is superior to H^2 is almost unnoticeable for the sharply imbalanced internal proportion (2-14 coupling) but becomes prominent as the internal proportion approaches balance. Consequently, it makes sense to use the additive function H^2 only if both: (i) there is a good reason to assume that the data contains mostly imbalanced coupled clusters and (ii) there is a reason to assume high level of noise. Real world data might be noisy, but given no explicit indication that the emerging configurations are inherently imbalanced, the multiplicative function H^3 is preferable. Consequently, we have used H^3 in our experiments with textual data, described in the following sections.

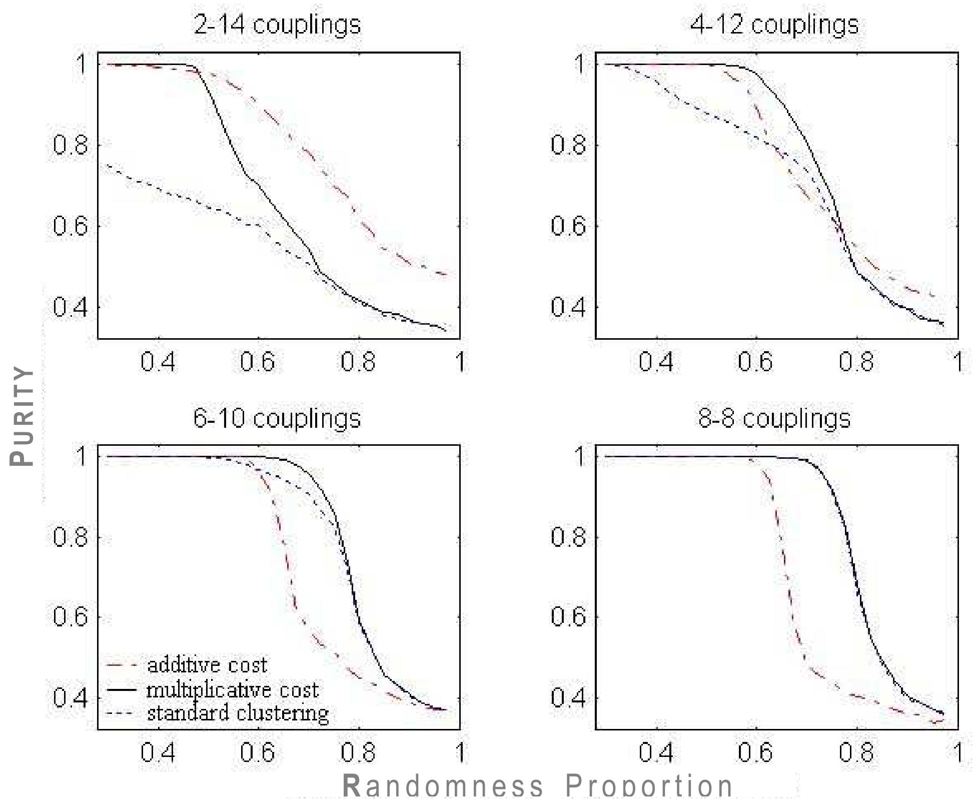


Figure 4.3: Purity as a function of the noise level (randomness proportion) for different coupled size proportions, obtained through experiments in reconstructing synthetic coupled-clustering configurations. For each proportion, results obtained using the straightforward adaptation of the original method (H^1, termed here "standard clustering"), "additive" (H^2) and "multiplicative" (H^3) cost functions are compared.

4.4 Identifying Corresponding Topics in Textual Corpora

In this section, we demonstrate the capabilities of the coupled clustering algorithm with respect to real-world textual data, namely pairs of sets of keywords (the subsets X^1, X^2 mentioned in Subsection 4.2.2) along with counts of co-occurring content words, taking the role of features. The keywords have been extracted from given corpora focused on distinct domains. Our experiments have been motivated by the target of identifying, by means of the induced coupled clusters, concepts that play similar or analogous roles in the examined domains. In Subsection 4.4.1, the keyword sets are extracted from collections of news articles referring to two conflicts of different character that are nowadays in the focus of public attention: the *Middle East conflict* and the dispute over electronic media copyright, demonstrated through the *Napster case*. Our experiments revealed some illuminating correspondences between the two seemingly unrelated conflicts. In Subsection 4.4.2, we turn to larger corpora focused on various religions, specifically *Buddhism*, *Christianity*, *Hinduism*, *Islam* and *Judaism*. Hence, the task is to explicate common, or equivalent, aspects of the examined religions. This inter-religion comparison is further analyzed and evaluated also in subsequent sections.

In both conflict and religion comparisons, our setting assumes that the datasets are given or that they can be extracted automatically. We have used the TextAnalyst software by MicroSystems Ltd.[3], which is capable of identifying key-phrases in given text, to generate datasets for our experiments. From the terms and phrases that have been identified by the software, we have excluded the items that have appeared in fewer than three documents. Thus, relatively rare terms and phrases that the software has inappropriately segmented have been filtered out.

After extracting the datasets, between-subset similarities, if not pre-given, are calculated. In general terms, every extracted keyword is represented by a co-occurrence vector, whose entries essentially correspond to the co-located words (concrete examples follow in the subsections bellow), excluding a limited list of function words. Then, between-subset similarity values are calculated using methods, such as those described in Subsection 4.1.2, to adapt the data for the similarity-based coupled-clustering algorithmic setting introduced in Section 4.2. We differentiate between two optional sources that can provide the co-occurrence data for the similarity calculations. One option is to base the calculations on co-occurrences within the same corpora from which the keyword sets have been extracted. Thus, the calculated similarity values naturally reflect the context in which the comparison is being made. This approach has underlay most of the coupled clustering experiments that we have conducted (Subsection 4.4.2). However, sometimes the compared corpora might be of small size and there is a need to rely on a more informative statistical source. An alternative option is to utilize the co-occurrences within an additional independent corpus for the required similarity calculations. In order to produce reliable and accurate similarity values, such independent corpus can be chosen to be significantly larger than the compared ones, but it is important that it addresses well the topics that are being compared, so the context reflected by the similarities is still relevant. This approach, making use of pre-given similarity values, is demonstrated in the following subsection.

4.4.1 Conflict Keyword Clustering Based on Pre-given Similarities

The conflict corpora are composed of about 30 news articles each (200–500 word tokens in every article), regarding the two above-mentioned conflicts: the Middle East conflict and the dispute over music copyright. The articles were downloaded in October 2000.

[3] An evaluation copy of TextAnalyst 2.3 is available for download at **http://www.megaputer.com/php/eval.php3**.

Table 4.1: Coupled clustering of conflict related keywords. Every row in the table contains the keywords of one coupled cluster. Cluster titles and titles of the three groups of clusters were added by the author.

	Middle-East	Music Copyright
Parties and Administration		
Establishments	**city state**	**company court industry university**
Negotiation	**delegation minister**	**committee panel**
Individuals	**partner refugee soldier terrorist**	**student**
Professionals	**diplomat leader**	**artist judge lawyer**
Issues and Resources in Dispute		
Locations	**home house street**	**block site**
Protection	**housing security**	**copyright service**
Activity and Procedure		
Resolution	**defeat election mandate meeting**	**decision**
Activities1	**assistance settlement**	**innovation program swap**
Activities2	**disarm extradite extradition face**	**use**
Confrontation	**attack**	**digital infringement label shut violation**
Communication	**declare meet**	**listen violate**
Poorly-clustered keywords		
low similarity values	**interview peace weapon**	**existing found infringe listening medium music song stream worldwide**
no similarity values	**armed diplomatic**	

We have obtained the similarities from a large body of word similarity values that have been calculated by Dekang Lin, independently of our project (Lin, 1998). Lin has applied the sim_L similarity measure (Subsection 4.2, Equation 4.9) to word co-occurrence statistics within syntactic relations, extracted from a very large news-article corpus.[4] We assume that this corpus includes sufficient representation of the conflict keyword sets in relevant contexts. That is: even if the articles in the corpus do not explicitly discuss the concrete conflicts, it is likely that they address similar issues, which are rather typical as news topics. In particular, occurrences of the clustered keywords within this corpus are assumed to denote meanings resembling their sense within our small article collection.

As Table 4.1 shows, the coupled-clusters that have been obtained by our algorithm fall, according to our classification, within three main categories: "Parties and Administration", "Issues and Resources

[4] This corpus contains 64 million word tokens from Wall Street Journal, San Jose Mercury, and AP Newswire. The similarity data is available at **http://armena.cs.ualberta.ca/lindek/downloads/sims.lsp.gz**.

in Dispute" and "Activities and Procedure". To improve readability, we have also added an individual title to each cluster.

The keywords labeled "poorly-clustered", at the bottom of Table 4.1, are assigned to a cluster with average similarity considerably lower than the other clusters, or for which no relevant between-subset similarities are found in Lin's similarity database. Consequently, these keywords could be straightforwardly filtered out. However, poorly clustered elements persistently occur in most of our experiments and we include them here for the sake of conveying the whole picture.

Making use of pre-given similarity data is, on the one hand, trivially advantageous. Apart from saving programming and computing resources, such similarity data typically relies on rich statistics and its quality is independently verified. Moreover: in principle, pre-given similarity data could be utilized for further experiments in clustering additional datasets that are adequately represented in the similarity database. However, there are several disadvantages in taking this route. First, reliable relevant similarity data is not always available. In addition, the context of comparing two particular domains might not be fully articulated within generic similarity data that has been extracted in a much broader context. For example, the interesting case where the same keyword appears in both clustered sets, but it is used for different meanings, could not be traced. A keyword used differently in distinct corpora would co-occur with different features in each corpus. In contrast, when similarities are computed from a unified corpus, self-similarity is generally equal to the highest possible value (1 in Lin's measure), which is typically much higher than other similarity values. In such case, the two distinct instances of a keyword presenting in both clustered sets would always fall within the same coupled-cluster.

4.4.2 Religion Keyword Clustering

This subsection introduces the main body of our data, to which, from this point on, the coupled clustering method is systematically applied, followed by detailed examination and evaluation of the outcome. The same data is further analyzed, in the next chapter, through additional algorithmic extensions. The data consists of five corpora, each focusing on a different religion: Buddhism, Christianity, Hinduism, Islam and Judaism, to which we apply our methods in order to compare the religions to one another and to identify corresponding aspects. The corpora used here significantly extend the ones used by Marx, et al. (2002).

As we have noted earlier, one of the options for inducing input similarity values is by using co-occurrence statistics from corpora that are focused on the compared domains. These can be the same corpora from which the clustered keywords are extracted. In such case, it is clear that each keyword appears in its relevant sense or senses. Hence, context dependent subtleties, such as identical keywords denoting different meanings, can be revealed. In this case, we rely on the assumption that there is a substantial overlap between the features, namely words commonly co-occurring in the two corpora, and that at least some of the overlapping features are used similarly within both. Specifically, we assume that the corpora to which we refer below – introductory web pages and encyclopedic entries concerning religions – contain enough common vocabulary directed towards some "average-level" reader, thus enabling co-occurrence-based similarity calculations that are fairly informative. In summary, while pre-given similarity data might typically result from richer statistics over a unified set of features, the alternative might fit better the context of the task at hand, but depends on rich enough statistics of shared features.

4.4.2.1 The Data

The religion data consists of five corpora containing encyclopedic entries, electronic periodicals and additional introductory web pages that were downloaded from the Internet. The five corpora contains 1.5–2 million word tokens (8.5–13 Megabyte) each. Using the TreeTagger[5] software, we have filtered out all function words according to their part-of-speech (POS) and substituted each one of the remaining words by its lemma. This way, each corpus has been shrunk to 0.8–1.2 million tokens (5.5–8.5 Megabyte). More details about the corpora can be found in Appendix A. In addition to the keywords extracted by TextAnalyst software (described above), the elements of the clustered sets include keywords that have been provided by comparative religion experts (the data provided by experts has been primarily used for quantitative evaluation, see Subsection 4.4.2.3). The total size of each of the final keyword sets is 180–240, of which 15–20% were not extracted by TextAnalyst but solely by the experts. Each keyword is represented by its co-occurrence vector, as extracted from its own corpus (so the same keyword that is relevant to two or more corpora has different representation with respect to each corpus). In counting co-occurrences, we have used two-sided sliding window of ±5 words, truncated by sentence ends (similarly to Smadja, 1993). On one hand, this window size captures most syntactic relations (Martin, Al & van Sterkenburg, 1983). On the other hand, this scope is wide enough to score terms that refer to the same topic in general – and not only literally interchangeable terms – as similar (Gorodetsky, 2001), which accords our aim of identifying corresponding topics. Appendix A contains details of some of the features that are most common in the corpora. The keyword sets are introduced through some of their items along with exemplifying features and corresponding co-occurrence counts.

Each one of the clustered keywords is represented by a (sparse) vector, whose entries are the counts of the keyword's co-occurrences with each feature. We have applied to the obtained vectorial

[5] TreeTagger – a language independent POS tagger and lemmatizer – is available for download from http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html.

representations the sim_{DMM} similarity measure, which incorporates detailed information on the data (Dagan, Marcus & Markovitch 1995; Subsection 4.1.2, Eq. 4.8; sim_L is less detailed in that it only distinguishes between features that are present in one of the two vectors: features that are present in both do not contribute to the dissimilarity even though the values might be different, while sim_{DMM} utilizes value differences for features that are positive in both vectors). After calculating between-subset similarities, we ran the coupled clustering algorithm on each pair of subsets.

4.4.2.2 Qualitative Overview of the Results

Appendix B contains a detailed sample of coupled-clustering results, with number of clusters $k = 16$, including four religion pairs: Buddhism–Christianity, Christianity–Hinduism, Hinduism–Islam and Islam–Judaism. The coupled keyword clusters are ordered by their average similarity in descending order. The poorly clustered elements – those in the 16th cluster with the lowest average similarity – are not shown. We have attached intuitive titles to each cluster for readability and orientation.

The following post-hoc thematic analysis combines careful examination of the results described in this section and some background reading (particularly Smart, 1989; see discussion in Section 4.5). The obtained clusters (and additional results that are not shown concerning other religion pairs) appear to reflect consistently several themes that share commonalities across the different subsets:

∗ The religious experience:

> This theme incorporates, for instance, terms referring to spiritual and mental conditions that lead to, or are the result of, the search for the religious message or religious belief.

∗ Theology and philosophy:

> This covers several aspects such as ethics and other basic principles to be followed. Two notable subtopics that are typically expressed through distinct clusters: qualities and attributes that are admired, usually related to the nature of the divine and, in contrast, the issues of sorrow, suffering, sin and punishment, usually adjoined with terms referring to the reward promised to those who do not violate the religious way.

∗ Institutional organizations:

> This topic covers the various schools and traditions within the religion and, sometimes, the history of their development. It includes, for instance, terms referring to various types of priests serving at the particular religion. It often involves names of places, where the various traditions have been originated, or are currently practiced.

∗ Practice and custom:

> This topic includes ritual aspects of the religion, for instance, terms referring to dietary rules, pilgrimage, holy places and festivals.

∗ The scriptures:

In addition to the names of holy writings, there are two notable sub-themes: terms referring to teaching and scholarship and terms related to myths and narratives. The latter sub-theme often involves names of figures and places that are related with the narratives.

Furthermore, keyword coupled clusters are a valuable source for additional specific analyses of varied sorts. We shall exemplify this here only briefly:

- **Religion founders.** The names of the central figures of each religion are often clustered together within the same coupled-cluster. Whenever such a cluster is not focused on other personal names, the cluster's terms often convey prominent attributes of the central figure, thus provide in some sense the key to the whole religion – the ideal attribute it adopts. Examples are *being*, *practice* and *teaching* with regard to Buddha, *believing, faith*, *love* regarding Jesus Christ and *messenger*, *prophet*, *Quran* regarding Mohamed.
- **Family relations.** Family related terms – *family*, *wife*, *husband*, *marriage*, *mother*, *sister*, *brother*, *daughter*, *child* (excluding *father* and *son*, which often convey additional meaning) – are distributed differently over clusters, in different pairs of religions. Islam plays a pivotal role with relation to these terms. Clustering Islam terms against those of Christianity and Judaism, the family terms are concentrated within a single coupled cluster. This provides a hint regarding the central part of family issues in the Islam, when it is viewed in light of the other western religions where comparable aspects are present (in contrast, comparing Christianity and Judaism with one another, the same terms are distributed among four different couple clusters). When Islam is compared to Buddhism, the family-related terms are divided among varied contexts: “personal relationships” (*enemy*, *fight*, *meet*, *responsibility*, …), “sin and prohibitions” (*forbid*, *kill*, *pain*, *punishment*, …), “figures in narratives” (*Abraham*, *Ishmael*, *Moses*, *caliph*, *tribe*, …).

We provide further qualitative evaluation of more results concerning comparison of religions in the following subsections.

4.4.2.3 Expert Data Used for Evaluation

As the previous section explicates, our empirical experiments have been concentrated on the comparative study of religions. In fact, the existence of such a discipline and its presumed potential as a source for external standards was an initial reason for applying our framework for religion comparison. The different religions are non-trivially related to one another. Thus, religions seem to be liable to varied types of analysis and views that might underlie interesting correspondences between them. Our working hypothesis is that corresponding aspects of distinct religions would be expressed through common features, i.e. commonly shared content words, implying close, or related, meaning. However, from the data contributed by the experts it also becomes apparent that there is no precise definition or consensual agreement with regard to the aspects of correspondence and the particular terms capturing them.

We have asked human subjects, whose academic field of expertise is the comparative study of religions, to perform manually the coupled clustering task in the following manner. The experts have been asked to explicate the most prominent equivalent aspects common to given pairs of religions. (To convey a broad notion of equivalency, we have included the following phrase in the instructions: "… features and aspects that are *similar*, or *resembling*, or *parallel*, or *equivalent*, or *analogous* in the two religions under examination…"; see Appendix C for the full instructions). Then, the experts have been requested to specify representative terms that characteristically address each one of the identified similar aspects, within the content world of the two compared religions. The resulting pairs of corresponding sets, or *classes*, of terms, each of which addressing one aspect of similarity between two compared religions, form our external standard – a *class configuration* (see Section 3.3). Such a configuration is used for evaluating our results regarding the particular religion pair to which it refers.

The task of explicating keywords, dissociated from any wider context, was new and somewhat unusual to our experts. We have made efforts not to add on top of that any bias that further restrictions might cause. We have provided only rough guidelines regarding the number or content of equivalent aspects (i.e. expert classes), and the number and identity of terms that are associated with each aspect within each religion (i.e. the size of the coupled clusters; see Appendix C). We have not set limits to the number of equivalent aspects with which any word can be associated.

We have got responses from four people that have accomplished the task – two graduate students and two university professors, from Finland, Israel and New Zealand. Perhaps due to the pretty permissive guidelines, we could not use some parts of the contributed data. There were several terms that did not occur, or occurred rarely (i.e. less than 40 times in the relevant corpus), in our corpora. A particular contribution, by one of the four experts, contained too few prevalent terms and thus has been discarded altogether, so we have been left with the expert class configurations contributed by the three remaining experts. One of the experts has provided comparisons between all 10 possible pairs of the five religions. Another expert has provided the following comparisons: Buddhism-Christianity Buddhism-Hinduism, Christianity-Islam and Christianity-Hinduism. The third expert has provided the three possible comparisons among Christianity, Islam and Judaism.

The data in use still included phrases conveying ideas that were far too composite than what we expected from key terms (e.g., the phrase *not-admiring-something-that-belongs-to-someone-else*). Most phrases of this type were excluded. With regard to few of them we made some editorial interpretations. For example (one of the most extreme cases): identifying the single expert phrase *obeying-God-and-not-mentioning-his-name-without-a-reason* with co-locations of the terms *obey* and *God*. Extensive work has been required also for other editing tasks, such as identifying alternating spellings. In total, we discarded from the expert classes about 50 (10%) of the terms contributed by the three experts, so that a total of 448 terms were left to be used in the evaluation. The detailed expert data is described further in Appendix C.

As noted, the field of comparative study of religions does not lend itself to an absolute measure for assessing keyword-grouping results, through the experts' contribution. Accordingly, our external standard cannot be regarded as conveying a definite academic directive. Nevertheless, we suppose that our appliance to domain experts have yielded data that is both more reliable and richer than what could have been collected from, say, "educated human subjects". Note that even for experts the task of identifying corresponding themes in religions is intrinsically subjective, most likely because their skills do not underlie clear-cut criteria. The fact that we have used data from three participants allows us to provide some notion of the maximal level of precision that we can, in principle, get.

4.4.2.4 Examples of Expert Data versus Coupled Clustering Output

Table 4.2 shows concrete examples for our results in comparison to the expert data used for evaluation. The top of the table (A) displays a specific coupled class configuration contributed by one of the experts, pertaining to Christianity to Hinduism. The expert configuration is followed by the output of our method. The next two configurations, in (B) and (C), have been produced by our coupled clustering algorithm, making use of the multiplicative H^3 and additive H^2 cost functions respectively. Although the expert configuration consists of five clusters, the most convincingly interpretable results, shown in the table have been obtained with eight clusters. The table demonstrates that reconstruction of the expert configuration follows, in several cases, the right direction, but it is still imperfect. There are several expert classes – e.g., the one titled "mysticism" – for which no trace is found in the various computerized outputs. On the other hand, computerized configurations display some level of topical coherence, unrelated with the expert clusters, for example, the cluster that we have titled *religious experience* in Table 4.2 (B), referring to the multiplicative function performance. The "one-to-many" coupled-clusters produced by the additive cost function (C) do reveal, as well, some interesting themes: symbols, doctrine, theological principles and so on. The themes captured, however, are not balanced well over the two religions and consequently do not overlap well with the expert classes, even in cases where there is some thematic correspondence. For comparison with standard single-set clustering, we used the IB method reviewed in the next chapter (Section 5.2). The standard clustering results in Table 4.2 (D), place all Hinduism terms in one cluster, which disallow the detection of any correspondence between the two religions.

Table 4.2: Examples of the expert data and coupled clustering results.

(A) Expert class configuration: The class configuration, comparing Christianity to Hinduism, contributed by expert I. The titles are those given by the expert.

Christianity	Hinduism
1. Scriptures	
new_testament old_testament apostle bible john luke matthew paul revelation	Gita mahabharata upanishad vedas
2. Beliefs and Ideas	
jesus_christ love_of_god devil god cross fish heaven hell resurrection trinity	holy_people trimurti moksha atman brahman reincarnation
3. Society and Politics	
catholic church minister monk priest protestant rome vatican	brahmin caste sadhu
4. Establishments	
bishop cardinal church pope priest	caste gift priest temple
5. Mysticism	
eucharist crucifixion love miracle saint suffer	ashram chakra darshan guru yoga

(B) Multiplicative cost function: Eight-cluster configuration produced by our program with the multiplicative function (the best score was obtained for this eight-cluster configuration, although the expert specified five classes). The eighth cluster, of lowest average intra-cluster similarity, is omitted from the multiplicative cost function results. The remaining seven clusters are shown in full. The results are shown in full, including terms not used by the expert. (Cluster titles are by the author).

Christianity	Hinduism
(1. religious experience)	
being believing child death earth faith father find **god**^{2} hear holy jesus **love**^{5} man people prayer problem sin soul spirit **suffer**^{5} word	being child death family find god human man people soul
(2. writings–1)	
america **bible**^{1} book **church**^{3,4} evangelical history religious **rome**^{3} theology tradition write	ancient art author book christian country history india language philosophy question religion religious sacred sanskrit school science shri south study **temple**^{4} tradition **vedas**^{1} west write
(theology)	
divinity doctrinal experience human moral relationship religion spiritual	animal attain **brahman**^{2} consciousness dharma discipline divinity existence experience faith freedom idea karma law liberation practice ritual sense shiva social society spirit spirituality teach universe word **yoga**^{5}
(writings–2)	
author chapter greek hebrew **luke**^{1} **matthew**^{1} **new_testament**^{1} **old_testament**^{1} passage **revelation**^{1} scripture study text theory translate writer writing	epic **gita**^{1} hymn literature **mahabharata**^{1} purana ramayana rigveda scripture story sutra teaching text **upanishad**^{1} writing
(doctrine / schools)	
ancient baptist **bishop**^{4} **catholic**^{3} constantinople convert council establish found german jew luther organization orthodox **pope**^{4} **protestant**^{3} university **vatican**^{3} west	aryan authority **brahmin**^{3} buddhism **caste**^{3,4} civilization doctrine found founder jain muslim scholar shaiva
(tradition / cutoms)	
christmas city disciple family friend home house jerusalem learn meet member **minister**^{3} ministry school service sunday woman worship	**ashram**^{5} ceremony dance festival ganesh **gift**^{4} holy krishna learn meditation pilgrimage prayer **priest**^{4} puja rama **sadhu**^{3} son star student teacher
(narratives)	
abraham angel **apostle**^{1} authority baptism baptize believer birth bless blood command confess **devil**^{2} eat eye face faithful fire flesh forgiveness gift gospel grant **heaven**^{2} **hell**^{2} holy_spirit israel **jesus_christ**^{2} **john**^{1} judgment kingdom law listen mankind moses mother **paul**^{1} pay peace preach prophet punishment question redemption refer repentance **resurrection**^{2} reward righteousness sabbath sacrifice **saint**^{5} sake salvation savior sinful sinner teach voice water win	birth devotee earth **guru**^{5} heaven mother person sacrifice

Table 4.2 (cont.): Examples of the expert data and coupled clustering results

(C) Additive cost function: Eight-cluster configuration produced by our program with the additive function. All clusters are displayd, but terms not used by the expert are shown only in the cases they are the only ones in their cell. Expert terms are in bold font. Superscripts inticate expert class number. (Cluster titles are by the author).

Christianity	Hinduism
(1. spirituality)	
spiritual	**guru**^{5} **yoga**^{5}
(2. religious)	
religious	**ashram**^{5} **brahmin**^{3} **caste**^{3,4} **priest**^{3,4} **temple**^{4}
(3. personal experience)	
apostle^{1} **bible**^{1} **devil**^{2} **god**^{2} **hell**^{2} **jesus_christ**^{2} **john**^{1} **love**^{5} **love_of_god**^{2} **paul**^{1} **resurrection**^{2} **suffer**^{5}	Person
(4. history – writings)	
history	**gita**^{1} **mahabharata**^{1} **upanishad**^{1} **vedas**^{1}
(5. establishments)	
church^{4} **minister**^{3} **saint**^{5}	Devotee
(6. symbols)	
cross^{2} **fish**^{2} **heaven**^{2} **miracle**^{5}	Heaven
(7. doctrine)	
bishop^{4} **cardinal**^{4} **catholic**^{3} **crucifixion**^{5} **eucharist**^{5} **luke**^{1} **matthew**^{1} **monk**^{3} **new_testament**^{1} **old_testament**^{1} **pope**^{4} **priest**^{3,4} **protestant**^{3} **revelation**^{1} **rome**^{3} **vatican**^{3}	doctrine
(8. theological principles)	
trinity^{2}	**atman**^{2} **brahman**^{2} **chakra**^{5} **darshan**^{5} **gift**^{4} **holy_people**^{2} **moksha**^{2} **reincarnation**^{2} **sadhu**^{3} **trimurti**^{2}

(D) Single set clustering: Eight-cluster configuration produced by a standard clustering method (the information bottleneck iterative algorithm, producing soft clusters (Section 5.2); each term is assigned into its most probable cluster). The Hinduism terms were all assigned in one cluster, so only Christianinity terms are shown. Exemplifying terms not used by the experts are shown only in the cases where there have been no expert terms in their cluster. Expert terms are in bold font. Superscripts indicate expert class number. (Cluster titles are by the author).

Christianity	Hinduism
(1. establishment-A)	
vatican^{3} **pope**^{4} **cardinal**^{4} **rome**^{3} **bishop**^{4} **catholic**^{3} **protestant**^{3}	
(2. customs/<general for Hinduism>)	
trade pilgrimage	**[All Hinduism terms were assigned to cluster 2]**
(3. spirituality)	
holy_spirit holy reign spirit	
(4. establishment-B)	
church^{4} **monk**^{3} **eucharist**^{5}	
(5. writings/figures)	
luke^{1} **matthew**^{1}	
(6. doctrine)	
postmodern theology luther evangelical ethic tradition religious religion founder	
(7. <general>)	
apostle^{1} **bible**^{1} **devil**^{2} **god**^{2} **hell**^{2} **jesus_christ**^{2} **john**^{1} **love**^{5} **love_of_god**^{2} **paul**^{1} **resurrection**^{2} **suffer**^{5} **minister**^{3} **saint**^{5} **cross**^{2} **fish**^{2} **heaven**^{2} **miracle**^{5} **crucifixion**^{5} **priest**^{3,4} **revelation**^{1} **trinity**^{2}	
(8. sacred writings)	
new_testament^{1} **old_testament**^{1}	

4.4.2.5 Quantification of the Overlap with the Expert Data

We employed the Jaccard coefficient to quantify the overlap between our output clusters and the classes provided by the experts. We used the version of Jaccard coefficient that is specifically adapted to the coupled clustering task, considering only cross-dataset element pairs (*JACCARD-PROB-CP*; see Chapter 3, Subsection 3.3.2.2). In order to lay common grounds for measuring the overlap, we eliminated from our clusters those terms that were not used by the expert. Note that the data was clustered in full, so that the terms not used by the expert were deleted from the outcome, *after* the clustering process was completed. This procedure differs from the one used by Marx et al. (2002), where the clustering algorithm was applied to partial datasets that included only expert's terms. Considering the noisy impact of those many items that are not in the target classes (about 400 items per couple of full datasets, compared to an average of 54 expert items used in each evaluation), the current procedure demonstrates a higher degree of robustness.

We compared coupled clustering results obtained with the multiplicative cost function H^3 to the additive function H^2, as well as to random assignments and to the clusters produced by standard clustering method – the *information bottleneck* iterative algorithm (Tishby, Pereira & Bialek, 1999; reviewed in Chapter 5, Section 5.2) – applied to the union of the two coupled subsets. The information bottleneck method produces soft probabilistic clustering, i.e., it assigns each element to all clusters with probabilistic assignment values that sum up to 1. We turned these probabilistic assignments into hard ones, by considering each element as if it is assigned into its most probable cluster. The original soft IB clusters can be evaluated through the methods we use as well, but in general, they score somewhat worse than the hard version.

Figure 4.4 displays the results for all 17 evaluation cases examined. High Jaccard coefficient values imply high degree of overlap with the expert classes. The number of clusters indicated by each expert, which is denoted in the figure by a dotted vertical line, does not perfectly predict the number of clusters that actually obtain the highest score, so it cannot be assumed known in advance. We rather examine output configurations with numbers of clusters that vary over a reasonable predetermined range: two to 16 clusters. Averaged over all shown cluster numbers across all religion pair cases, the differences between the methods are all statistically significant, except for the difference between the additive cost function and the standard single-set clustering. Particularly, Figure 4.4 exhibits the superiority of the multiplicative cost in the vast majority of cases, over the whole cluster number range. In some cases, additionally to the highest scoring configuration, there are local picks (maxima) along the result graph, indicating that there is more than one meaningful interpretation to the data, corresponding to different levels of detail.

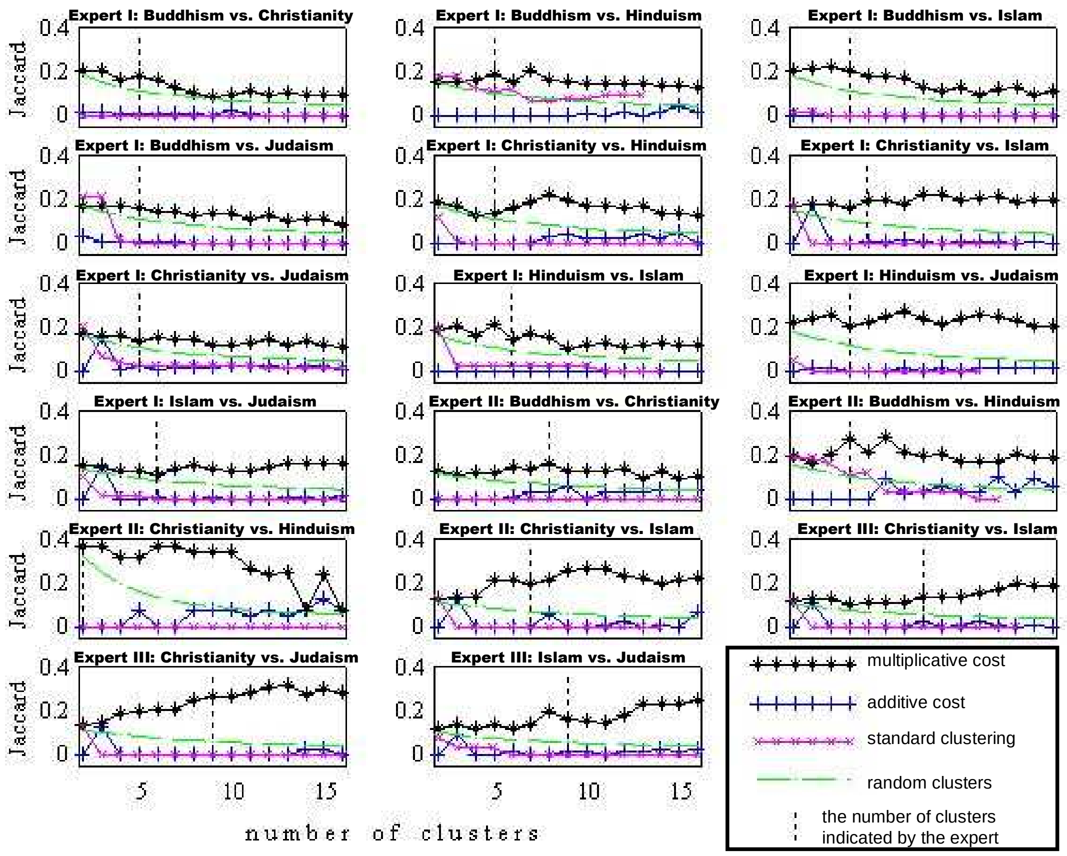


Figure 4.4: The religion keyword coupled-clustering results evaluated relatively to the expert classes. Jaccard scores are shown for cluster numbers range from two to 16, for all 17 cases: ten by expert I, four by expert II and three by expert III. The methods in use: coupled clustering with the multiplicative and additive cost functions, and a standard clustering method (information bottleneck). Scores of random assignments are shown as well.

In most cases, the additive-cost and standard-clustering Jaccard scores shown in Figure 4.4 lie below the corresponding random-assignment scores. The reason is that the version of Jaccard coefficient used here considers only cross-dataset element pairs (Chapter 3, Subsection 3.3.2.2). The additive cost function tends to form "one-to-many" coupled clusters, i.e., clusters that contain only one, or very few, elements from one of the datasets (for an example, see Table 4.2 (C)). Standard clustering, as well, tends to follow within-dataset regularities, inducing similarly imbalanced clusters.

4.4.2.6 Agreement between the Experts

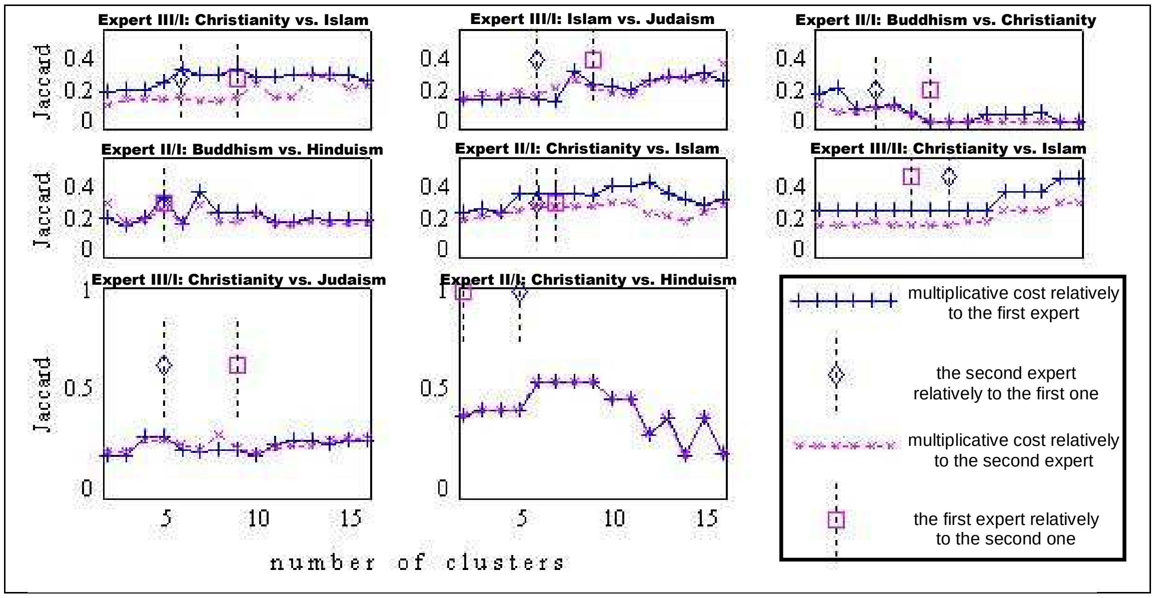


Figure 4.5: The religion keyword coupled-clustering results evaluated on partial sets of terms: those used by two experts for the same cross religion comparison. The Jaccard scores of the 10 cases are shown for all cluster numbers from two to 16, three common to experts I and II, three common to experts I and III and one common to experts II and III. For comparison, we show the level of agreement between the experts, i.e., the Jaccard score each expert configuration achieves in approximating the classes provided by the other expert.

In this subsection, we quantify the subjectivity level that can be ascribed to the evaluation criterion in use and we examine the portion of our results comparable with the limits set by this subjectivity level. There was one religion pair (Christianity/Islam) to which all three experts generated evaluation data independently and five additional religion pairs to which two experts generated data independently (Buddhism/Christianity, Buddhism/Hinduism and Christianity/Hinduism by expert I and II; Christianity/Judaism and Islam/Judaism by experts I and III). Together, evaluating data of an expert against data regarding the same religion pair contributed by another expert gave a total of 16 evaluation cross-expert evaluation cases: ten cases resulting from the religion pairs addressed by two experts and six cases from the pair addressed by all experts, as this religion pair was actually addressed by three pairs of experts. Note that each expert pair was judged twice, taking one expert as a gold standard and the other expert as the one being evaluated.

Evaluating expert data through comparing it to data of another expert measures cross-expert overlap over the set of terms used in common by both experts, thus such evaluation results provide an indication for the level of agreement between the experts. Figure 4.5 displays the Jaccard scores indicating these cross-expert agreement levels for each pair of religions, along with the results obtained by our method on the same small sets of commonly used terms. In order to set common grounds to our results with cross-expert agreement, after the clusters were formed the terms not used by *either* expert were discarded from our clusters, so that the term sets left are much smaller than the ones used for evaluation in the previous subsection. The figure shows that our results approach the cross-expert agreement level in some of the cases. However, the averaged results in table 4.4 show that even the best clustering results for each case (over the range of number of clusters) are significantly inferior to the agreement between experts. In the next chapter, these results would be significantly improved.

Table 4.4: Quantitative measures for cross-expert agreement, obtained through applying the evaluation methods to expert-classes using another expert class configuration as a criterion. For comparison, we show mean scores of our results with respect to the 16 corresponding religion comparison cases, to which an additional expert has referred. The evaluation is restricted to the items common to both experts. In parentheses: the average over the best score of each case.

	Jaccard Coefficient
Means ± standard deviations **for cross-expert agreement quantitative assessment**	
Cross-expert Agreement	**0.450**±0.249
Means ± standard deviations **over the 16 cross-expert scores averaged** (best) **over all examined numbers of clusters 2–16**	
Multiplicative Cost	**0.237**±0.101 (0.344±0.114)
Difference: Expert – Multiplicative	**0.214**±0.194 (0.106±0.193)

4.5 Discussion

In this chapter we have formalized and implemented the coupled clustering problem that was introduced in general terms in the previous chapter: clustering two pre-given element subsets to matching parts so that each matched pair forms a coupled cluster. Formalization of the task took place in the familiar pairwise cost-based data clustering framework (Subsection 4.2.2). The implementation has used the stochastic Gibbs Sampler search method (Subsection 4.2.4). The requirement of matching the formed subset parts has been realized through restricting the pairwise clustering setting to only those similarities between members of distinct subsets (Subsection 4.2.1).

The results demonstrate that our approach addresses the coupled clustering task fairly well, not only with respect to tailored synthetic task (Section 4.3), but also for tackling an interesting real-world problem (Section 4.4). Neither standard clustering techniques nor simplistic approach, such as the one suggested by the straightforward additive cost function (Eq. 4.11), address the examined task as well as the solution dedicatedly designed for the problem, namely the multiplicative cost function (Eq. 4.14).

The expertise of the individuals that participated in creating our evaluation criteria did not completely eliminate the subjectivity inherent to the task of identifying and matching terms related with various religions. Given the inherently subjective task and the lack of clear-cut criteria for matching equivalent terms or themes within the studied data, we find the results encouraging and inherently not too far from the level of agreement among the experts.

Yet, several aspects in the method that we have introduced seem as non-negligible limitations. There is a source of information, the between-subset similarities, which are not utilized at all. Should they really play no role in the formed correspondence between systems aligned with respect to each other? Another point to note is that the conversion from element-feature data to pairwise similarities (through methods as the ones described in Subsection 4.1.2) implies additional loss of information.

In the next chapter of this work, we introduce another method that generalizes the coupled clustering setting across several aspects and, in addition, addresses the points we have mentioned.

Chapter 5: Cross-partition Clustering

In this chapter, we move on from the elementary setting of the coupled-clustering approach described in the previous chapter to a more general approach, termed *cross-partition* (CP) clustering. Section 5.1 below details the differences between coupled clustering and the CP framework. After a detailed review in Section 5.2 of methods grounding our approach, Section 5.3 introduces the CP method. The method is demonstrated experimentally in Section 5.4 and is discussed further in the concluding Section 5.5.

5.1 Cross Partition versus Coupled Clustering

The CP setting generalizes some aspects of the problem treated by the coupled-clustering method. The aspects by which cross-partition clustering differs from the coupled-clustering method are:

- Cross-partition clustering is in general “soft”. A data element can be assigned to several clusters, in varying assignment levels, at the same time. Specifically, the assignments are probabilistic, i.e., the assignment levels of any given element associating it with all clusters are all non-negative and sum up to 1 (see Chapter 2, Subsection 2.1.2.2).

- The coupled-clustering framework models analogies through producing clusters that contain elements of two distinct subsets of the data. However, other than the convention of thinking about analogies as involving two systems, there is no inherent reason for restricting the setting to two subsets. The cross-partition clustering setting allows pre-given partitioning of the data element set to more than two subsets, across which *correspondences* are to be drawn (talking about ‘correspondence’ might seem more appropriate than ‘analogy’ for this generalized setting).

- Formally, the cross-partition approach allows also ‘soft’ pre-partitions: elements can be probabilistically assigned to some or all of the pre-given subsets (which should not be confused with probabilistic assignments to clusters). Our experimental work, however, was restricted to the hard pre-partition setting.

- Another characteristic attribute of the coupled-clustering framework is that it assumes given pairwise similarity values that apply to pairs of elements of the two pre-given subsets. While in principle data might happen to be readily available in this form, our practical experience was with data consisting of co-occurrence counts of data elements with features. Co-occurrence counts can provide basis for calculating pairwise similarities (see Chapter 2, Subsection 2.1.3.4 and Chapter 4, Subsection 4.1.2), but they can be utilized also more directly. The cross-partition clustering setting processes element-feature count distributions rather than pairwise similarities. Thus the mediating stage of computing similarities, which necessarily implies loss of information, is avoided.

- One last noticeable aspect of the coupled clustering method is that it ignores the whole collection of within-subset similarities altogether. The cross-partition method tackles the neutralization of within-subset regularities in a more principled manner.

5.2 Background: Information Theoretic Approaches

The CP method takes an information-theoretic approach to data clustering, which is related with the "communicative" aspect of data clustering (see Chapter 2, Subsection 2.1.1). Particularly, we elaborate on the *information bottleneck* data clustering method (IB, Tishby, Pereira & Bialek, 1999; Gilad-Bachrach, Navot & Tishby, 2003) and on an earlier variation on the same distributional-clustering theme (Pereira, Tishby & Lee, 1993), which has been recently studied further under the name *Information Distortion* clustering (ID, Gedeon, Parker & Dimitrov, 2003).

In Section 5.2.1, we discuss the ID method, which is taken as the basis for our elaboration, with an emphasis on the role of the maximum entropy principle (Jaynes, 1982) in this method. As the two methods are tightly related, some results obtained originally with regard to the IB method are cited as well. In Section 5.2.2, we refer more specifically to the IB method, which underlies additional variants of our CP algorithm. Both methods are described as aiming at minimization problems (rather than constrained minimization or other types of optimization). One further development around the IB theme, directly relevant to the CP task, is the method of *information bottleneck with side information* (IB-SI, Tishby & Chechik, 2003), reviewed in Section 5.2.3.

5.2.1 The Information Distortion Method

The ID method was introduced, under the name *distributional clustering*, by Pereira, Tishby & Lee (1993) and has recently been studied further by Gedeon, Parker & Dimitrov (2003).

5.2.1.1 Input and Output

As a probabilistic clustering method (see Chapter 2, Subsection 2.1.4.6), the ID method employs formal random variables X, Y and C with values ranging over all data elements, features and cluster labels, respectively. The relative frequency $p(x)$ of each data element x to occur in the given dataset and the conditional probabilities $p(y|x)$ of each feature y to occur in association with each element x are given as input. Based on this input, the ID method outputs a probability distribution $p(c|x)$ over the clusters for each element x. This distribution defines x's "assignment level" or "association level" with each cluster c. In addition, the ID method constructs conditional probability distributions, $p(y|c)$, over all features for every cluster *c*. The $p(y|c)$ distributions can be considered as supplementary output, specifying a representative for each cluster *c*, a centroid in the feature space (Subsection 2.1.4.4).

5.2.1.2 Underlying Principles and Formulation

The ID method designates the relevance features as the sole basis for directing the formation of clusters: *clusters are formed so that they are optimally informative with regard to the feature distribution*. In order to determine the assignments of data elements into the formed clusters, the ID method applies also the maximum entropy principle (Jaynes, 1982) constrained by the first direction.

Many optimization problems (including data clustering, see Chapter 2, Subsection 2.1.4.2) are formulated so that they are solvable through minimization of a *cost term* or a *Lyapunov function* (often, the minimum practically obtained is local, implying sub-optimal solution). The ID method, as well, accomplishes the counterbalance between the two above principles, feature relevance and maximum entropy, through minimizing a single cost term – *the ID functional*:

$$F^{ID} \equiv -H(C|X) + \beta \hat{H}(Y|C) . \qquad (5.1)$$

$\beta > 0$ is a parameter counterbalancing the relative impact of the two principles. $H(C|X)$ is the *entropy* of cluster distribution conditioned on the distribution of data elements

$$H(C|X) \equiv -\sum_{c,x} p(c,x) \log p(c|x) = -\sum_x p(x) \sum_c p(c|x) \log p(c|x) , \qquad (5.2)$$

where *x* and *c* range over all possible values of the variables *X* and *C*, i.e., the sum runs over all cluster-element combinations. $\hat{H}(Y|C)$ is defined as

$$\hat{H}(Y|C) \equiv -\sum_{c,x} p(c,x) \sum_y p(y|x) \log p(y|c) = -\sum_x p(x) \sum_c p(c|x) \sum_y p(y|x) \log p(y|c) . \qquad (5.3)$$

The conditional entropy $H(C|X)$ is the expected length of a transmission, communicating that *C* has the value *c* under the assumption that the value of *X* is known to be *x*. It quantifies the overall information that the data variable *X* leaves unexplained with regard to the cluster variable *C*, or in other words, the level of uncertainty regarding cluster distribution knowing the data distribution (see Thomas & Cover, 1991, p. 20)[1]. Following the maximum entropy principle, the goal of the ID method is to maximize this uncertainty expressed by $H(C|X)$ (subject to the constraint). Accordingly, the ID method seeks to *minimize* the F^{ID} term, which counterbalances *minus* $H(C|X)$ against $\hat{H}(Y|C)$.

[1] In general, the definition in Eq. (5.3) employs base 2 logarithm. However, as a change in the logarithm base adds a constant to the conditional entropy and other related values, we prefer to use throughout this chapter the natural log, which somewhat simplifies mathematical derivations.

The $\hat{H}(Y|C)$ term introduces the constraint of forming clusters that are informative with regard to the features ($\hat{H}(Y|C) = 0$, for instance, implies that the clusters completely determine the feature distribution). It incorporates an expected value of $p(y|c)$ distributions, averaged over the feature distribution $p(y|x)$ relatively to each data element x.

The ID method follows one further assumption that we term here *the ID conditional independence assumption* (also known as the *markovity assumption*, Gilad-Bachrach, Navot & Tishby, 2003), stating that clusters and features are assumed independent given the data:

$$p(c,y|x) \quad = \quad p(c|x)\, p(y|x) \tag{5.4}$$

for each x, c and y. (Equivalently, one may require that clusters and features would share zero mutual information given the data: $I(C;Y\,|X) = 0$.[2]) Taking the expected value over all x of both sides of Eq. 5.4, we get

$$p(c,y) = \textstyle\sum_x p(x)\, p(c,y|x) = \sum_x p(x)\, p(c|x)\, p(y|x)\,. \tag{5.5}$$

It follows that under the conditional independence assumption, $\hat{H}(Y|C)$ is exactly equal to the entropy $H(Y|C)$ of feature distribution conditioned on the clusters:

$$\begin{aligned} H(Y|C) \quad &\equiv \quad -\textstyle\sum_{c,y} p(c,y) \log p(y|c) \quad = \\ &-\textstyle\sum_{c,y} (\, \sum_x p(x)\, p(c|x)\, p(y|x)\,) \log p(y|c) \quad = \quad \hat{H}(Y|C)\,, \end{aligned} \tag{5.6}$$

where c and y range over all possible values of the variables C and Y, i.e., the sum runs over all cluster-feature combinations. Therefore, if the independence assumption holds (which turns to be the case, as shown in the next subsection), we can rewrite F^{ID} (Eq. 5.1) as

$$L^{ID} \quad = \quad -H(C|X) + \beta H(Y|C)\,. \tag{5.7}$$

In conclusion, given that the independence assumption is satisfied, the ID method maintains a counterbalance between maximizing $H(C|X)$, to keep high uncertainty level regarding assignments into clusters, and minimizing $H(Y|C)$. $H(Y|C)$ measures the uncertainty about the feature distribution

[2] The explicit formula for the equivalent form of the conditional independence assumption is:

$$I(C;Y|X) \;=\; \sum_x p(x) \sum_{c,y} p(c,y\,|\,x) \log \frac{p(c,y\,|\,x)}{p(c\,|\,x)\,p(y\,|\,x)} \;=\; 0\,.$$

If for all c, x and y $p(c,y|x) = p(c|x)\, p(y|x)$ ($\neq 0$) then the arguments of all log terms in the sum are equal to 1 and hence $I(C;Y|X) = 0$. Suppose now $I(C;Y|X) = 0$. Mutual information amounts to a sum of *KL* divergences, each of which is non-negative and is equal to zero if and only if its arguments are identical distributions (Cover & Thomas, 1991 p. 19), i.e., $p(c,y|x) = p(c|x)\, p(y|x)$ for all c, x and y.

left after revealing cluster distribution and, therefore, minimizing $H(Y|C)$ realizes the principle with which this subsection opens: clusters are expected to be informative about feature distribution. As explained, the ID method follows this principle restrictedly: minimizing F^{ID} indeed decreases the above uncertainty so that formed clusters are informative with regard to the feature distribution, but only up to the level enabled by the maximum-entropy directed element assignments.

5.2.1.3 The ID Algorithm

The iterative ID algorithm was originally introduced by Pereira, Tishby & Lee (1993). The algorithm consists of two steps that update the $p(c|x)$ and $p(y|c)$ distributions, each in its turn, so that they accomplish the weighed balance between minimizing $H(Y|C)$ and maximizing $H(C|X)$, through consistent decrease of the F^{ID} value.

```
Set t = 0, and repeatedly iterate the two update-steps sequence below, till convergence (at time step
t = 0, initialize p_t(c|x) randomly or arbitrarily and skip step ID1):
    ID1:  $p_t(c|x) = \frac{1}{z_t(x,\beta)} e^{-\beta KL\left[p(y|x) \| p_{t-1}(y|c)\right]}$
              where  $z_t(x,\beta) = \sum_{c'} e^{-\beta KL\left[p(y|x) \| p_{t-1}(y|c')\right]}$
    ID2:  $p_t(y|c) = \frac{1}{p_t(c)} \sum_x p(x) p_t(c|x) p(y|x)$
              where  $p_t(c) = \sum_x p(x) p_t(c|x)$
    $t = t + 1$
```

Figure 5.1: The iterative ID clustering algorithm (with fixed β, and $|C|$).

At the starting iterative cycle, when $t = 0$, the ID1 step just initializes the $p(c|x)$ distributions randomly or arbitrarily. At all later cycles (with $t > 0$), step ID1 updates all $p(c|x)$ values leaving $p(y|c)$ unchanged, so that the value of F^{ID} (Eq. 5.1) decreases as the following lemma shows. The second part of this lemma affirms that step ID2, which updates the $p(y|c)$ values leaving $p(c|x)$ fixed, decreases the value of F^{ID} as well.

Lemma 5.1:

(A) At any iterative cycle with $t > 0$, update step ID1 decreases the value of F^{ID} by

$$\Delta F^{ID1}{}_{t} \;=\; \textstyle\sum_{x} p(x) KL[p_{t-1}(c|x) \| p_{t}(c|x)] \,. \tag{5.8}$$

(B) Update step ID2 decreases the value of F^{ID} by

$$\Delta F^{ID2}{}_{t} \;=\; \beta \textstyle\sum_{c} p_{t}(c) KL[p_{t}(y|c) \| p_{t-1}(y|c)] \,. \tag{5.9}$$

Proof: see Appendix D (the proof of part (A) is original; (B) follows Gilad-Bachrach, Navot & Tishby, 2003))

Note that step ID2 affects only one of the components of F^{ID}, namely $\hat{H}(Y|C)$, as the other component of F^{ID}, $H(C|X)$, does not depend on $p(y|c)$. Step ID2 imposes the conditional independence assumption (Eq. 5.5). Therefore, at the end of each iterative cycle also the alternative formulation of F^{ID} (Eq. 5.7) is guaranteed to decrease (ID1 is only assured to decrease the value of F^{ID} as given in Eq. 5.1, as the independence assumption does not hold).

Lemma 5.2 (following Tishby, Pereira & Bialek, 1999)**:** Stable points of the ID algorithm (i.e., probability distributions that remain unchanged under the update steps, so that $p_{t+1}(c|x) = p_t(c|x)$ and $p_{t+1}(y|c) = p_t(y|c)$ for all c, x and y) are local extremum points of F^{ID} (Eq. 5.1).

Proof: see Appendix D

Conclusion: The ID algorithm converges to a local minimum of F^{ID} (unless initialized to an extremal point of a different type).

Proof: From lemma 5.1, the value of F^{ID} decreases in each iterative cycle of the ID algorithm (with $t > 0$) by a non-negative quantity $\Delta F^{ID}{}_{t} = \Delta F^{ID1}{}_{t} + \Delta F^{ID2}{}_{t}$. As $-H(C|X) \geq -H(C) \geq -\log|C|$ and $\beta H(Y|C) \geq 0$, the value of F^{ID} is bounded from below and the algorithm converges to a locally minimal value (unless initialized to a stable value that is not minimal). From Lemma 5.2 it follows that those probability distributions, $p(c|x)$ for each x and $p(y|c)$ for each c, assigning to F^{ID} its stable value at the ID algorithm convergence point[3] define an extremal, hence (locally) minimal, point of F^{ID}.

The ID algorithm is a version of the *k*-means scheme described in Chapter 2 (Subsection 2.1.4.4). Step ID1 assigns each element to each cluster in proportion to their similarity in the feature space as

[3] Gilad-Bachrach, Navot & Tishby (2003) show that, assuming the number of local minima is finite, the convergence is onto particular definite limit distributions (otherwise, it could have been the case that the sequence of distributions assigning the converging sequence of values to F^{ID} do not converge).

calculated in the previous update cycle. More concretely, each assignment $p_t(c|x)$ is in inverse proportion to the *KL* divergence between the feature vector representation of x ($p(y|x)$) and the centroid of c as calculated in the last iterative cycle ($p_{t-1}(y|c)$). The *KL* divergence is not an arbitrary dissimilarity measure, even though the general *k*-means scheme allows such arbitrariness. Rather, the *KL* divergence emerges from the ID cost term, so that it is particularly tailored to address the considerations underlying this cost. Step ID2 updates the $p_t(y|c)$ distributions so that they satisfy the conditional independence assumption and they are consistent with the input and the recently calculated $p_t(c|x)$ distributions.

5.2.1.4 Controlling the Number of Clusters by Modifying the Value of β

The value of the parameter β governs the tendency of the re-assignments performed by step ID1 to be probabilistic or deterministic. With $\beta = 0$, implying that only the $H(C|X)$ part of F^{ID} is articulated, assignments of each element x to all clusters c are in equal probability (so all clusters are in fact identical) as the unconstrained maximum entropy principle entails. For larger β values, assignments turn more discriminative. In the limit of $\beta \rightarrow \infty$, each element is assigned, with probability 1, to a distinct singleton (assuming the number of clusters $|C|$ is allowed to be as large as the number of data elements $|X|$ and unless there are elements with identical feature representations).

In between the above two extreme cases of zero and infinity, the value of β dictates the number of distinct clusters that can be formed in a manner resembling thermodynamics of physical systems, where β takes a role that is opposite to that of temperature. The higher β is, i.e., the stronger is the bias to construct clusters that convey detailed information regarding feature distribution, a larger number of distinct clusters is enabled. Specifically, for any given number of clusters $|C| = 2, 3, \ldots$, there is a minimal β value enabling the formation of $|C|$ distinct clusters. Setting β to be smaller than this critical value corresponding to the current $|C|$ would result in two or more duplications of the same cluster. Once β is raised just above the critical value, the same cluster would not duplicate any more: it splits, or *bifurcates*, to two distinct clusters due to the stronger emphasis on the requirement to convey feature information.

Based on the above dynamics, the iterative algorithm can be applied repeatedly within a gradual cooling-like, or *deterministic annealing*, scheme: starting with random initialization of the $p_0(c|x)$'s, generate two clusters, to be discovered empirically, with the critical β value for $|C| = 2$. Then, use a perturbation on the obtained two-cluster configuration to initialize the $p_0(c|x)$'s for a larger set of clusters and execute additional runs of the algorithm to identify the critical β value for the larger $|C|$. And so on: each output configuration is used as a basis for a more granular one. In our actual experiments, we always split one cluster – the largest one (of highest $p(c)$) – so each output configuration includes one cluster more than its predecessor. The final outcome is a “soft hierarchy” of probabilistic clusters.

5.2.2 The Information Bottleneck Method

The IB method interprets clustering as a *distorted representation*, optimized for conveying the meaningful part of the information embodied within given data. In their presentation, Tishby, Pereira & Bialek (1999) base the IB method on the notion of *mutual information* rather than the conditional entropy we use. They define the IB cost term to be minimized (the *IB functional*) as

$$L^{IB} = I(C;X) - \beta I(Y;C). \tag{5.10}$$

As $I(C;X) = H(C) - H(C|X)$ and $I(Y,C) = H(Y) - H(Y|C)$, it turns that F^{IB} closely resembles the ID cost term F^{ID} (Eq. 5.7). The two terms differ by subtraction of $\beta H(Y)$, which is a constant depending on β and the data, and by addition of $H(C)$ that is not a constant factor. Note that taking Eq. 5.10 as the IB cost term presumes an independence assumption, the same as in the ID method (Eqs. 5.4, 5.5). Gilad-Bachrach, Navot & Tishby (2003) explicate a Lyapunov function (corresponding to Eq. 5.1) that does not depend on this assumption.

As the IB and ID cost terms resemble each other, also the iterative IB algorithm that finds a local minimum for F^{IB} is similar to the ID algorithm (Figure 5.1):

Set $t = 0$, and repeatedly iterate the three update-steps sequence below, till convergence (at time step $t = 0$, initialize $p_t(c|x)$ randomly or arbitrarily and skip step ID1):

```
IB1:  $p_t(c|x) = \frac{1}{z_t(x,\beta)} p_{t-1}(c) e^{-\beta KL[p(y|x) \| p_{t-1}(y|c)]}$
          where $z_t(x,\beta) = \sum_{c'} p_{t-1}(c') e^{-\beta KL[p(y|x) \| p_{t-1}(y|c')]}$
IB2:  $p_t(c) = \sum_x p(x) p_t(c|x)$
IB3:  $p_t(y|c) = \frac{1}{p_t(c)} \sum_x p(x) p_t(c|x) p(y|x)$
$t = t + 1$
```

Figure 5.2: The iterative IB clustering algorithm (with fixed β, and $|C|$).

There are two differences between the IB algorithm and the ID algorithm. The IB algorithm includes a separate step for calculating $p_t(c)$. In the ID algorithm, the same calculation actually takes place but $p_t(c)$ has the mere role of a normalization factor. The other difference is that a prior of $p_{t-1}(c)$ is added to the term calculated in update step IB1. We hence occasionally refer to the IB algorithm as a *priored* version of the ID algorithm, and to the ID algorithm as a *non-priored* version of the IB algorithm.

Lemma 5.3: The update cycle at time t decreases the value of L^{IB} (Eq. 5.10) by $\Delta F^{IB1}{}_t + \Delta F^{IB2}{}_t + \Delta F^{IB3}{}_t$, where

$$\begin{aligned}
&\text{(A) } \Delta F^{IB1}{}_t &=& \textstyle\sum_x p(x) KL[p_{t-1}(c|x)\|p_t(c|x)]\,, \\
&\text{(B) } \Delta F^{B2}{}_t &=& KL[p_t(c)\|p_{t-1}(c)]\,, \qquad (5.11) \\
&\text{(C) } \Delta F^{B3}{}_t &=& \beta \textstyle\sum_x p_t(c) KL[p_t(y|c)\|p_{t-1}(y|c)]\,.
\end{aligned}$$

Proof: Minimizing L^{IB} is equivalent, under the appropriate independence assumption (Eq. 5.5), with minimizing the following term

$$F^{IB} \equiv H(C) - H(C|X) + \beta \hat{H}(Y|C)\,. \qquad (5.12)$$

($\hat{H}$ is as in the definition in Eq. 5.3). It can be shown that step IB1 decreases F^{IB} by $\Delta F^{IB1}{}_t$ and step IB-3 decreases it by $\Delta F^{IB3}{}_t$, following the same argumentation as in Lemma 5.1 (A) and lemma 5.1 (B), respectively. A proof that step IB2 decreases F^{IB} by $\Delta F^{IB2}{}_t$, which relies on argumentation similar to that of the proof of lemma 5.1 (B), is given by Gilad-Bachrach, Navot & Tishby (2003).

Lemma 5.4 (Tishby, Pereira & Bialek, 1999)**:** Stable points of the IB algorithm are locally extremal points of F^{IB} (Eq. 5.10).

Proof: The same idea as in the proof of Lemma 5.2 above (proved in Appendix D).

From Lemmas 5.3 and 5.4, proof of convergence for the IB algorithm follows, as in the convergence proof of the ID algorithm (Subsection 5.2.1.3).[4]

5.2.2.1 The IB Method and Information Theory

The IB method draws an illuminative relation between data clustering and Claude Shannon's information theory. Rate-distortion theory (Thomas & Cover, 1991, ch. 13) shows that the average of number of bits needed for conveying a distorted (lossy) representation C of information X is the mutual information $I(C;X)$. (Minimizing this mutual information is equivalent with *maximizing* $H(X|C) = H(X) - I(C;X)$, which is the number of bits that are *saved* due to lossy encoding being employed, out of the $H(X)$ bits that are needed to represent X with no loss). The complementary

[4] In the same vein, for any cost term $-H(C|X) + \alpha H(C) + \beta H(Y|C)$, with positive α and β, there is an algorithm similar to the IB algorithm minimizing it. The modification required in order that the IB algorithm will work in this general case is replacing, in step IB1, the prior $p_{t-1}(c)$ with $p_{t-1}(c)^{\alpha}$. Then, Lemma 5.3 holds, with (B) replaced by $\Delta F^{B2}{}_t = \alpha KL[p_t(c)\|p_{t-1}(c)]$.

constraint of the IB method, maximizing $I(C;Y)$, which is completely equivalent with minimizing $H(Y|C)$ as done by the ID method, is related with another topic in information theory – the channel-capacity problem (Thomas & Cover, 1991, pp. 190–194). By combining these two classical problems into one doubly constrained problem, the IB method interprets data clustering as minimizing data representation size, liable to preserving the part that is most informative with regard to the features.

5.2.3 Information Bottleneck with Side Information

Recently, Chechik & Tishby (2003) introduced the method of information bottleneck with *side information* (IB-SI). Their approach emerged from recognizing that production of relevant clusters can be facilitated through considering attributes of the data that are *irrelevant* to the patterns to be revealed, in distinction from the standard relevance features. In order to incorporate the effect of these additional attributes, the IB-SI method introduces an additional set of *irrelevance features* represented by a new variable Y^{-}.

The IB-SI method, like the IB and ID methods, aims at minimizing a cost term. Specifically, the cost term to be minimized by the IB-SI method is:

$$L^{IB\text{-}SI} = I(C;X) - \beta I(Y^{+};C) + \gamma I(Y^{-};C)\,. \tag{5.13}$$

This term incorporates the impact of the irrelevance features Y^{-} as if it symmetrically opposes the bias introduced by the relevance features (represented here by Y^{+}, rather than by Y). As in the derivation of the IB and ID algorithms, an iterative algorithm can be based on the IB-SI cost term:

Set $t = 0$, and repeatedly iterate the four update-steps sequence below, till convergence (at time step $t = 0$, initialize $p_t(c|x)$ randomly or arbitrarily and skip step IB-SI1):

$$\text{IB-SI1:}\quad p_t(c|x) = \frac{1}{z_t(x,\beta)} p_{t-1}(c) e^{-\beta\left(KL[p(y^{+}|x)\|p_{t-1}(y^{+}|c)] - KL[p(y^{-}|x)\|p_{t-1}(y^{-}|c)]\right)}$$

$$\text{where}\quad z_t(x,\beta) = \sum\nolimits_{c'} p_{t-1}(c') e^{-\beta\left(KL[p(y^{+}|x)\|p_{t-1}(y^{+}|c')] - KL[p(y^{-}|x)\|p_{t-1}(y^{-}|c')]\right)}$$

$$\text{IB-SI2:}\quad p_t(c) = \sum\nolimits_x p(x) p_t(c|x)$$

$$\text{IB-SI3:}\quad p_t(y^{+}|c) = \frac{1}{p_t(c)} \sum_x p(x) p_t(c|x) p(y^{+}|x)$$

$$\text{IB-SI4:}\quad p_t(y^{-}|c) = \frac{1}{p_t(c)} \sum_x p(x) p_t(c|x) p(y^{-}|x)$$

$$t = t + 1$$

Figure 5.3: The IB-SI clustering iterative algorithm (with fixed β, γ and $|C|$).

The two lemmas below are the IB-SI equivalents of the ID and IB lemmas (Subsections 5.2.1.3, 5.2.2).

Lemma 5.5: The update cycle at time t subtracts from the value of $L^{IB\text{-}SI}$ (Eq. 5.13) $\Delta F^{SI1}{}_t + \Delta F^{SI2}{}_t + \Delta F^{SI3}{}_t - \Delta F^{SI4}{}_t$, where

$$\begin{aligned} \text{(A) } \Delta F^{SI1}{}_t &= \textstyle\sum_x p(x) KL[p_{t-1}(c|x) \| p_t(c|x)]\,, \\ \text{(B) } \Delta F^{SI2}{}_t &= KL[p_t(c) \| p_{t-1}(c)]\,, \\ \text{(C) } \Delta F^{SI3}{}_t &= \beta \textstyle\sum_x p_t(c) KL[p_t(y^+|c) \| p_{t-1}(y^+|c)]\,, \\ \text{(D) } \Delta F^{SI4}{}_t &= \gamma \textstyle\sum_x p_t(c) KL[p_t(y^-|c) \| p_{t-1}(y^-|c)]\,. \end{aligned} \tag{5.14}$$

Proof: Minimizing $F^{IB\text{-}SI}$ is equivalent, under the appropriate independence assumption (as in Eq. 5.5, incorporating Y^-, symmetrically to Y^+), to minimizing the following term

$$F^{IB\text{-}SI} \equiv H(C) - H(C|X) + \beta \hat{H}(Y^+|C) - \gamma \hat{H}(Y^-|C)\,. \tag{5.15}$$

($\hat{H}$ is as in the definition in Eq. 5.3). Following argumentation similar to that of Lemmas 5.1 and 5.3, it can be shown that step IB-SI1 decreases $L^{IB\text{-}SI}$ by $\Delta F^{SI1}{}_t$, step IB-SI2 decreases it by $\Delta F^{SI2}{}_t$, step IB-SI3 decreases it by $\Delta F^{SI3}{}_t$ and step IB-SI2 *increases* it by $\Delta F^{SI4}{}_t$.

Lemma 5.6: Stable points of the IB-SI algorithm are extremal points of $F^{IB\text{-}SI}$ (Eq. 5.13).

Proof: Following the same argumentation as in Lemmas 5.2 and 5.4 above.

However, the argumentation used for proving the convergence of the ID and IB algorithms (Subsections 5.2.1.3, 5.2.2) cannot be applied in the IB-SI case. Convergence of the IB-SI algorithm depends on the ratio between $\Delta F^{SI1}{}_t + \Delta F^{SI2}{}_t + \Delta F^{SI3}{}_t$ and $\Delta F^{SI4}{}_t$, thus cannot be proven for any arbitrary combination of β, γ, $|C|$ and a given dataset.

The IB-SI approach extends the IB method, thus it facilitates explaining clustering with side information in classical information theoretical terms. A slightly different approach to clustering with side information is based on the ID method. The underlying cost-term of this ID-based variant is

$$L^{ID\text{-}SI} = -H(C|X) + \beta H(Y^+|C) - \gamma H(Y^-|C)\,. \tag{5.16}$$

From this cost term the following iterative algorithm is derived:

```
Set $t = 0$, and repeatedly iterate the three update-steps sequence below, till convergence (at time step $t = 0$, initialize $p_t(c|x)$ randomly or arbitrarily and skip step ID-SI1):

    ID-SI1:   $p_t(c|x) = \frac{1}{z_t(x,\beta)} e^{-\beta\left(KL[p(y^+|x)\| p_{t-1}(y^+|c)] - KL[p(y^-|x)\| p_{t-1}(y^-|c)]\right)}$
              where $z_t(x,\beta) = \sum_{c'} e^{-\beta\left(KL[p(y^+|x)\| p_{t-1}(y^+|c')] - KL[p(y^-|x)\| p_{t-1}(y^-|c')]\right)}$

    ID-SI2:   $p_t(y^+|c) = \frac{1}{p_t(c)} \sum_x p(x) p_t(c|x) p(y^+|x)$
              where $p_t(c) = \sum_x p(x) p_t(c|x)$

    ID-SI3:   $p_t(y^-|c) = \frac{1}{p_t(c)} \sum_x p(x) p_t(c|x) p(y^-|x)$
              where $p_t(c)$ is as above

    $t = t + 1$
```

Figure 5.4: The ID-SI clustering iterative algorithm (with fixed β, γ and $|C|$).

Observations equivalent to the ones made above with regard to the IB-SI algorithm (Lemmas 5.5, 5.6) similarly hold with regard to the ID-SI algorithm.

5.3 The Cross-partition Method

Cross-partition (CP) data clustering aims at identifying, through clusters of data elements, themes that are common to several subsets that together form the given dataset. To this end, the formed clusters should *cut across* the pre-given partition into subsets: each cluster is expected to contain elements from all subsets. As mentioned, the CP problem generalizes the coupled-clustering setting of the previous chapter (the noticeable differences between the methods are detailed in Section 5.1 above). The basic setting is described in Chapter 3.

In this section, we introduce a novel approach to the CP clustering task. This task is particularly challenging in cases where the given subsets are relatively homogenous, i.e., the elements within each subset are typically more similar to one another compared to their similarity to elements of other subsets. The suggested method is designed to overcome such cases. Lead by feature information that is shared across the subsets, it produces clusters that capture the commonalities while neutralizing possibly salient within-subset regularities. Our method is inspired by the ID and IB methods reviewed above. Below, we describe how the CP method extends the ID data clustering setting

(Subsection 5.3.1). Then, we characterize the desired form of solution to this task (Subsection 5.3.2) and present the algorithm that we propose in order to accomplish it (Subsection 5.3.3). Finally, we specify additional versions of our algorithm, motivated by the differences between the IB and ID methods (Subsection 5.3.4).

5.3.1 The Cross-partition Data Clustering Task

The CP method, which addresses the CP data clustering task introduced in Chapter 3, extends the standard probabilistic clustering setting in terms of both input accepted and output constructs being produced, as described below.

5.3.1.1 Input: The Pre-partitioning Variable

The identity of the pre-given subset to which a particular data element belongs is a source of information that plays a role in the CP clustering task, additional to the relevance feature distribution. In order to articulate this information, we introduce an additional formal variable W, the *pre-partitioning variable*. The values that W can get range over the labels of the subsets of the pre-given partition (two or more subsets). We denote that a data element x belongs to a subset w, by writing $p(w|x) = 1$. If x does not belong to w, we write $p(w|x) = 0$. In our experiments (Section 5.4), we have restricted each element to be uniquely associated with one subset. Allowing $p(w|x)$ values between 0 and 1 would enable probabilistic ("soft") pre-partitioning, which accords with our formalism but have not been empirically tested. The pre-partitioning information $p(w|x)$ is supplemented to $p(x)$ and $p(y|x)$ as input considered by the CP method.

We note that in order that the CP method produces meaningful results, the pre-partitioning variable W is expected to correlate to some extent with the feature variable Y (i.e., $I(Y;W) > 0$, or equivalently $H(Y) > H(Y|W)$), additionally to the correlation between X and Y that is essential also for standard data clustering.

5.3.1.2 Output: Re-association of Features and Clusters

A common way to convey the essence of a cluster c in the probabilistic setting is to specify those elements x with highest $p(c|x)$ scores. It is interesting, as well, to specify in addition the features that are most typical to a cluster. We note in Chapter 2 that the centroid of a cluster c – the $p(y|c)$ feature distribution – indicates the location of the cluster in the feature space. However, the features that are most characteristic for a cluster c are those features y with high $p(c|y)$ scores rather than high $p(y|c)$, as the latter might reflect the fact that the feature y appears frequently in all clusters and not discriminatively in c.

In the ID (or IB) setting, $p(c|y)$ can be straightforwardly calculated through Bayes rule: $p(c|y) = p(y|c)\,p(c)\,/\,p(y)$. A novel aspect of the CP method is that it quantifies differently the level of association of features with clusters. Hence, along with the probability distributions $p(c|x)$, the CP method outputs distributions that associate each feature y to each one of the clusters c. We denote these probabilities by $p^*(c|y)$, to emphasize the fact that they are different than the $p(c|y)$ distributions of the ID case.

As in the ID case, the CP method produces representative probability distributions of features for each cluster. These representative distributions are derived from the newly introduced $p^*(c|y)$, hence denoted $p^*(y|c)$. Finally, the CP method produces yet another type of supplementary output, which has no correspondence in the ID and IB methods: for each combination of a cluster c and a pre-given subset w, a probability distribution over the features $p(y|c,w)$. Such distributions form feature-based centroids of c restricted to the elements originated in w or, as we term them, *W-projected centroids*.

5.3.2 Underlying Principles Characterizing the Solution

As stated above, there are four types of probability distributions that together form the CP method output: $p(c|x)$, which can be considered as the main target of the method, and in addition $p(y|c,w)$, $p^*(c|y)$ and $p^*(y|c)$ (where c, x, y and w denote cluster labels, data elements, features and pre-given subset labels, respectively). These four types of distributions constitute the whole set of parameters that the CP method manipulates.

The core idea of the CP method lies in the step implementing feature-cluster re-association conveyed through the $p^*(c|y)$ distributions (see Subsection 5.3.2.3 below). The associations of the characterizing features with the formed clusters are biased so that these associations become *independent* of the given pre-partition of the data. The conception and formulation of this imposed independence, underlying the focusing on relevant cross-system information and the defocusing of irrelevant system-specific information, is the basis to our original interpretation of the notion of analogy.

Below, we characterize in detail how all four types of probability distributions link up together as a solution to the CP clustering task.

5.3.2.1 Assignments of Elements to Clusters

Similarly to the case with the ID and IB methods, the assignments of elements to clusters in the CP method follow a maximum entropy principle. This is formalized through the following term:

$$F^{CP1} \;\equiv\; -H(C|X) + \beta \hat{H}^*(Y|C)\,, \tag{5.17}$$

where $\beta > 0$ is a counterbalancing parameter, $\hat{H}^*(Y|C)$ is defined to be

$$\hat{H}^*(Y|C) \equiv -\textstyle\sum_{c,x} p(c,x) \sum_y p(y|x) \log p^*(y|c) = -\sum_x p(x) \sum_c p(c|x) \sum_y p(y|x) \log p^*(y|c) \qquad (5.18)$$

and $H(C|X)$ is the entropy of cluster distribution conditioned on the data (Eq. 5.2), the constrained value of which the CP method seeks to maximize. The target of the CP method is thus to find $p(c|x)$ values, constrained to sum up to 1 for each x, which bring the value of F^{CP1} to minimum. Following this direction, the CP method maximizes $H(C|X)$, subject to a further constraint involving the probability distributions $p^*(y|c)$. For the purpose of assigning elements to clusters, the $p^*(y|c)$ distributions are considered as if they are given and fixed. We will refer later to how the CP method modifies these distributions (Subsection 5.3.2.4).

5.3.2.2 *W*-projected Centroids

As the case is with the IB and ID methods, the CP method aims at identifying a partition of the data that is optimally informative about the relevance features, represented by the variable Y. Such configuration may consider as an information source relevant to predicting feature distribution not only the partition to clusters, C, but also the pre-given partition W. Consequently, optimizing the feature information extractable from the two partitions together would be carried out through minimizing the conditional entropy term $H(Y|C,W)$. To be more precise, the CP method actually minimizes a related term, which is equivalent, under an appropriate independence assumption (explicated below), to $H(Y|C,W)$:

$$F^{CP2} \equiv \textstyle\sum_x p(x) \sum_c p(c|x) \sum_y p(y|x) \sum_w p(w|x) \log p(y|c,w)\,. \qquad (5.19)$$

The conditional independence assumption of the IB and ID methods (Eq. 5.4), is extended by the CP method to apply to W as well, namely C, Y and W are independent given X:

$$p(c,y,w|x) = p(c|x)\, p(y|x)\, p(w|x) \qquad (5.20)$$

(for each c, x, y and w), or equivalently $I(C;Y;W|X) = 0$.

Summing up both sides of Eq. 5.20 over all x values, we obtain

$$p(c,y,w) = \textstyle\sum_x p(x)\, p(c|x)\, p(y|x)\, p(w|x)\,. \qquad (5.21)$$

Assuming Eq. 5.21 holds, we can re-write F^{CP2}:

$$F^{CP2} = \textstyle\sum_{c,y,w} \log p(y|c,w) \sum_x p(x)\, p(c|x)\, p(y|x)\, p(w|x) = \qquad (5.22)$$

$$\textstyle\sum_{c,y,w} \log p(y|c,w)\, p(c,y,w) = H(Y|C,W)\,.$$

As we will see in Subsection 5.3.3, the independence assumption is indeed maintained by the CP method. Therefore, from Eq. 5.22 we conclude that by minimizing F^{CP2} the CP method minimizes the conditional entropy term $H(Y|C,W)$ or, in other words, it optimizes the level of information about the features provided by the combination of the clusters C and the pre-given partition W.

5.3.2.3 Feature-cluster Re-association

As said above (in Subsection 5.3.1.2), an innovative aspect of the CP approach is that it re-associates features with clusters differently than what is straightforwardly expected from the assignments of data elements into clusters. Re-associating features with clusters is carried out so that the associations reflect the following fundamental principle:

> *The way features (Y) and clusters (C) are associated is not supposed to correlate with the pre-partition (W).*

Assuming a triply joint probability distribution $p^*(c,y,w)$ (where the asterisk comes to distinguish between this probability to the one in Eq. 5.21), the above principle would be formulated as:

$$P^*(c,y,w) \;=\; p^*(c,y)\,p(w) \qquad (5.23)$$

(for each c, y and w), or equivalently $I^*(C,Y;W) = 0$.

Under the assumption that Eq. 5.23 holds, we formulate below a second maximum entropy principle that the solution to the CP problem is supposed to realize, which would direct the re-association of features with clusters. Specifically, the CP method aims at minimizing the following term:

$$F^{CP^*1} \;\equiv\; -\,H^*(C|Y) + \eta\,\hat{H}(Y|C,W)\,, \qquad (5.24)$$

where, $H^*(C|Y)$ is a conditional entropy term of cluster distribution conditioned on the distribution of features

$$H^*(C|Y) \;\equiv\; -\,\textstyle\sum_{c,y}\; p^*(c,y)\log p^*(c|y) \;=\; -\sum_y p(y)\,\sum_c p^*(c|y)\log p^*(c|y) \qquad (5.25)$$

(the sum runs over all cluster-feature combinations), and $\hat{H}(Y|C,W)$ is defined to be

$$\hat{H}(Y|C,W) \;\equiv\; \textstyle\sum_{c,y,w} p^*(c,y,w)\log p(y|c,w) \;=\; \sum_w p(w)\,\sum_y p(y)\,\sum_c p^*(c|y)\log p(y|c,w)\,. \qquad (5.26)$$

η is a parameter with a positive value, counterbalancing the relative impact of the two components of F^{CP^*1}. $\hat{H}(Y|C,W)$ articulates the constraint on the maximum entropy principle posed by the w-projected centroids. The target of the CP method is to find feature-cluster probabilistic associations $p^*(c|y)$, minimizing F^{CP^*1} while being constrained to sum up to 1 for each y. Thus, the CP method maximizes the conditional entropy $H^*(C|Y)$, subject to a further constraint posed by the probability distributions $p(y|c,w)$. Although we have already seen in the previous subsection how the $p(y|c,w)$ distributions are to be determined, for the purpose of re-associating features with clusters these constraining distributions are referred to as if they are given and fixed.

5.3.2.4 Centroids that Cut Across the Pre-partition

Finally, there are the centroids distributions used in characterizing the assignments of the solution to the CP problem (Subsection 5.3.2.1). These distributions are expected to minimize the following term:

$$F^{CP*2} = \sum_y p(y) \sum_c p^*(c|y) \log p^*(y|c)\,, \qquad (5.27)$$

which is identical to the conditional entropy $H^*(Y|C)$, as $p(y)\,p^*(c|y) = p^*(c,y)$. However, the $p^*(y|c)$ values are referred by F^{CP*2} as variables, while $p^*(c|y)$ are treated as if they are given and fixed.

5.3.3 The CP Algorithm

Starting from a random or arbitrary clustering configuration, the CP algorithm updates iteratively the four types of probability distributions $p(c|x)$, $p(y|c,w)$, $p^*(c|y)$ and $p^*(y|c)$. The algorithm's iterative update cycle follows the four principles described in the previous subsection. Each step of the cycle optimizes one class of probability distributions relatively to one of the above principles, while the other distributions are held constant.

Set $t = 0$ and repeatedly iterate the following update steps sequence, till convergence (in the first iteration, when $t = 0$ randomly or arbitrarily initialize $p_t(c|x)$ and skip step CP1):

CP1: $$p_t(c|x) = \frac{1}{z_t(x,\beta)} e^{-\beta KL\left[p(y|x)\,||\,p^*_{t-1}(y|c)\right]}$$

where $z_t(x,\beta) = \sum_{c'} e^{-\beta KL\left[p(y|x)\,||\,p^*_{t-1}(y|c')\right]}$

CP2: $$p_t(y|c,w) = \frac{1}{p_t(c,w)} \sum_x p(x)\,p_t(c\,|\,x)\,p(y\,|\,x)\,p(w\,|\,x)$$

where $p_t(c,w) = \sum_x p(x)\,p_t(c\,|\,x)\,p(w\,|\,x)$

CP*1: $$p^*_t(c|y) = \frac{1}{z^*_t(y,\eta)} \prod_w p_t(y\,|\,c,w)^{\eta p(w)}$$

where $z^*_t(y,\eta) = \sum_{c'} \prod_w p_t(y\,|\,c',w)^{\eta p(w)}$

CP*2: $$p^*_t(y|c) = \frac{1}{p^*_t(c)} \sum_y p(y)\,p^*_t(c\,|\,y)$$

where $p^*_t(c) = \sum_y p(y)\,p^*_t(c\,|\,y)$

$t = t + 1$

Figure 5.5: The cross partition clustering iterative algorithm (with fixed β, η, and $|C|$).

The CP method probabilistically associates, or assigns, elements to clusters in proportion to the element-centroid similarity (step CP1 of the algorithm; in this respect, the CP method follows the *probabilistic representative-based clustering scheme,* Chapter 2, Subsection 2.1.4.6). More specifically, as in the IB and ID algorithms, an element x is assigned into a cluster c in proportion exponentially inverse to the KL divergence between the representative feature distributions $p(y|x)$ and $p^*_{t-1}(y|c)$. The KL divergence is not an arbitrary proximity measure but it rather emerges from the first maximum-entropy principle described in 5.3.2.1 above (see also Lemma 5.8 below).

Based on the assignments calculated by step CP1, the next step, CP2, calculates expected values of the current W-projected centroid distributions, $p_t(y|c,w)$. This step conforms to the information maximization direction of 5.3.2.2 above. Particularly, step CP2 imposes the extended conditional independence assumption (Eqs. 5.20, 5.21).

Step CP*1 re-associates features with clusters, by calculating for every feature y probability distribution over the clusters $p^*_t(c|y)$ proportional to biased ('flattened') geometric mean over all current $p_{t-1}(y|c,w)$ values. This step facilitates feature-cluster associations that cut across the pre-partitioned subsets: strong association of a feature y with a cluster c, i.e., a high $p^*_t(c|y)$ value, requires high $p_t(y|c,w)$ values across all subsets w (in contrast to some y' and c' for which $p_t(y'|c',w)$ is high on average but varies across the w's). The bias introduced within this weighted geometric-mean scheme is directed by a free parameter η. Low values of η underlie loss of information: $\eta = 0$ implies that all features are uniformly assigned to all clusters regardless of the $p_t(y|c,w)$ values. Higher η values preserve more of the information embodied within the W-projected centroids. Regardless of the value of η, step CP*1 integrates $p_t(y|c,w)$ values over all values of W, so the result is independent of any particular w. This scheme, which was motivated intuitively in Dagan, Marx & Shamir (2002), turns to realize the supplementary maximum-entropy direction introduced in Subsection 5.3.2.3.

Step CP*2 derives the $p^*_t(y|c)$ probability distributions, which are the centroids for the next update cycle, from the current $p^*_t(c|y)$ values and the input $p(y)$ distribution through Bayes rule. It realizes the information-maximization principle of Subsection 5.3.2.4.

In the case of the ID and IB algorithms, the existence of a cost term, that gets a smaller value at each update step, ensures the convergence to a configuration locally minimizing the value of the corresponding term. For the CP method, we cannot specify a cost term that is reduced by each update step, or by the whole update cycle. The algorithm, however, empirically converges in most examined test cases, particularly for all real-world and synthetic datasets where it has been reasonable to assume an underlying cross-partition structure (see Section 5.4 below). Whenever the algorithm converges, the resulting stable-point probability distributions $p(c|x)$, $p(y|c,w)$, $p^*(c|y)$ and $p^*(y|c)$ necessarily maintain the relations between the distributions explicated in Subsection 5.3.2.

5.3.3.1 Further Observations

Below, we show that each one of the CP algorithm update steps “improves” the currently given configuration relatively to the principle corresponding to this step (unless the current configuration is a stable point of the algorithm). This is done by reducing the term associated with that step relatively to the particular distributions that the step updates (this, however, does not imply that the CP algorithm iterative cycle reduces the value of any *single* cost term).

Lemma 5.7: In the update cycle of time t, the four CP algorithm update steps CP1, CP2, CP*1, CP*2, decrease the values of F^{CP1} (Eq. 5.17), F^{CP2} (Eq. 5.19), $F^{\mathrm{CP*1}}$ (Eq. 5.24), $F^{\mathrm{CP*2}}$ (Eq. 5.27) by

$$\begin{aligned}
&\text{(A) } \Delta F^{CP1}{}_{t} &=& \textstyle\sum_x p(x) KL[p_{t-1}(c|x) \| p_t(c|x)]\,, \\
&\text{(B) } \Delta F^{CP2}{}_{t} &=& \textstyle\sum_x p_t(c,w) KL[p_t(y|c,w) \| p_{t-1}(y|c,w)]\,, \\
&\text{(C) } \Delta F^{\mathrm{CP*1}}{}_{t} &=& \textstyle\sum_y p(y) KL[p^*_{t-1}(c|y) \| p^*_t(c|y)]\,, \\
&\text{(D) } \Delta F^{\mathrm{CP*2}}{}_{t} &=& \textstyle\sum_y p^*_t(c) KL[p^*_t(y|c) \| p^*_{t-1}(y|c)]\,,
\end{aligned} \qquad (5.28)$$

respectively.

Proof: see Appendix D

Lemma 5.8: A set of probability distributions that form a stable point of the CP algorithm (i.e., ones satisfying $p_{t+1}(c|x) = p_t(c|x)$, $p_{t+1}(y|c,w) = p_t(y|c,w)$, $p^*_{t+1}(c|y) = p^*_t(c|y)$ and $p^*_{t+1}(y|c) = p^*_t(y|c)$, for all c, x, y and w) specifies locally extremal points for: F^{CP1} with respect to $p(c|x)$ ($p^*(y|c)$ held fixed), F^{CP2} with respect to $p(y|c,w)$ ($p(c|x)$ held fixed), $F^{\mathrm{CP*1}}$ with respect to $p^*(c|y)$ ($p(y|c,w)$ held fixed) and $F^{\mathrm{CP*2}}$ with respect $p^*(y|c)$ ($p^*(c|y)$ held fixed).

Proof: see Appendix D

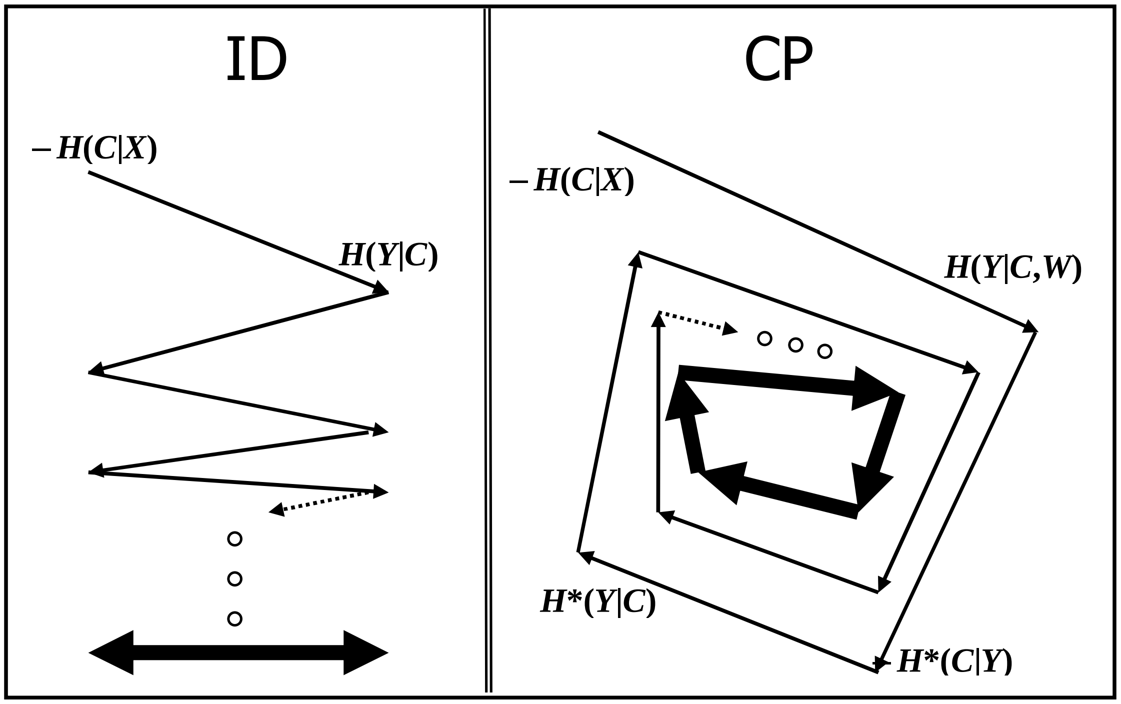


Figure 5.6: A schematic illustration of the dynamics of the ID algorithm versus that of the CP algorithm. In the ID algorithm, convergence is onto a configuration where the two systems of parameters complementarily balance one another, bringing a cost term to a locally minimal value. In the CP algorithm, stable configurations maintain balanced inter-dependencies (*equilibrium*) of four distinct systems of parameters.

During the execution of the CP algorithm, each of the four probability distribution types, $p(c|x)$, $p(c|y,w)$, $p^*(c|y)$ and $p^*(y|c)$, directs the formation of distributions of another type. In the resulting solution, the four types of conditional probability distributions take part in a closed cycle of dependencies, as described in Subsection 5.3.2. The dynamics characterizing the CP algorithm, in comparison to that of the ID algorithm, is illustrated in Figure 5.6.

Argumentation as used in the ID and IB cases cannot be used for proving convergence of the CP algorithm. Lemmas 5.7 and 5.8 do not provide information regarding how each of F^{CP1}, F^{CP2}, $F^{\mathrm{CP*1}}$ and $F^{\mathrm{CP*2}}$ is affected by the changes that occur in practice in factors that are considered to be fixed by the principle underlying its modification.

5.3.3.2 The Parameters β and η

Gradual increase of the value of β works in practice for the CP method much the same as it works for the IB and ID methods (Subsection 5.2.1.4): increasing β along subsequent runs enables the formation of configurations of growing numbers of clusters, each initialized based on a configuration of fewer clusters obtained previously. In general, we have experimented with η values that are fixed during a whole cycle of runs, while only β is gradually incremented in order to produce increasing number of clusters. Understanding better the role of η and the inter-dependencies between the given data, η and β would be an interesting topic for future research.

There are two cases where the scheme of incrementing β gradually while η is held fixed, in order to produce a growing number of clusters, seems not to work. First, we encountered cases of synthetic datasets (randomly drawn with no underlying pre-tailored cross-partition structure) where the algorithm eventually did not converge but rather went through an endless oscillatory pattern. This behavior, characterized further in Subsection 5.4.1.3, took place for restricted ranges of η values. Convergence was always obtained for some η values outside these ranges. Further, for those datasets where underlying cross-partition either existed by construction or was expected to exist based on the content of the data (as in the experimental work Section 5.4) the algorithm converged with no exception.

The other potentially problematic scenario is where the CP algorithm converges to fewer clusters than initialized: some clusters are gradually vanished as update cycles keep being performed. Note that this never happens in the iterative IB and ID algorithms, where the formation of a centroid (step ID2/IB3) implies that there is some mass of data elements concentrated around it, ensuring that some points would be reassigned to the corresponding cluster in the next re-assignment step (ID1/IB1). In contrast, the formation of a CP centroid (step CP*2) does not guarantee that the centroid is backed with enough mass of data elements from all subsets. As a result, it might happen that the dominant features in a centroid formed in the previous update cycle are not sufficiently weighty in one or more of the subsets and hence the relative total weight of the cluster might tend to zero as the iterations are carried on.[5] This behavior was observed in a variety of cases. Particularly for very detailed pre-partitons (high $|W|$), we were not able to produce even small numbers of clusters. On the other hand, in the $|W| \leq 5$ cases to which the experimental part of this work was restricted, setting a lower η value whenever such behavior occurred consistently lead to the formation of the desired number of clusters. The effect of $|W|$ on the behavior of the algorithm (in interaction with other factors) should be studied further, both experimentally and theoretically.

[5] Somewhat related to this might be our empirical observation that the IB/ID iterative algorithms, although formally guaranteed to converge, often produce small clusters that do not capture significant themes in the data.

5.3.4 CP Algorithmic Variations Inspired by the IB Method

The IB method reviewed in Subsection 5.2.2 minimizes the cost term L^{IB} (Eq. 5.10), which, up to a constant factor, can be re-written as $H(C) - H(C|X) + \beta H(Y|C)$. As already noted, this term is reminiscent of the ID method cost term $L^{ID} = -H(C|X) + \beta H(Y|C)$ (Eq. 5.7). The difference lies in the non-constant term $H(C)$. In a like manner, it is possible to modify the two maximum-entropy-based terms of the equations underlying the CP method, namely F^{CP1} and F^{CP*1} (Eqs. 5.17 and 5.24). Thus, we may replace F^{CP1} by

$$F^{CP1'} \equiv H(C) - H(C|X) + \beta \hat{H}^*(Y|C) . \qquad (5.29)$$

Regardless of the modification explicated by Eq. 5.29, we can also replace F^{CP*1} by

$$F^{CP*1'} \equiv H^*(C) - H^*(C|Y) + \eta \hat{H}(Y|C,W) , \qquad (5.30)$$

where $H(C)$ and $H^*(C)$ are the entropy of C based on $p(c)$ and $p^*(c)$ respectively.

Set $t = 0$ and repeatedly iterate the following update steps sequence, till convergence (in the first iteration, when $t = 0$ randomly or arbitrarily initialize $p_t(c|x)$ and $p^*_t(c)$ and skip step CP1):

CP1': $p_t(c|x) = \dfrac{1}{z_t(x,\beta)} p_{t-1}(c) e^{-\beta KL\left[p(y|x) \| p^*_{t-1}(y|c)\right]}$

where $z_t(x,\beta) = \sum_{c'} p_{t-1}(c') e^{-\beta KL\left[p(y|x) \| p^*_{t-1}(y|c')\right]}$

CP2': $p_t(c) = \sum_x p(x) p_t(c \mid x)$

CP3': $p_t(y|c,w) = \dfrac{1}{p_t(c,w)} \sum_x p(x) p_t(c \mid x) p(y \mid x) p(w \mid x)$

where $p_t(c,w) = \sum_x p(x) p_t(c \mid x) p(w \mid x)$

CP*1': $p^*_t(c|y) = \dfrac{1}{z^*_t(y,\eta)} p_{t-1}{}^*(c) \prod_w p(y \mid c, w)^{\eta p(w)}$

where $z^*_t(y,\eta) = \sum_{c'} p_{t-1}{}^*(c') \prod_w p_t(y \mid c', w)^{\eta p(w)}$

CP*2': $p^*_t(c) = \sum_y p(y) p^*_t(c \mid y)$

CP*3': $p^*_t(y|c) = \dfrac{1}{p_t{}^*(c)} p(y) p^*_t(c \mid y)$

$t = t + 1$

Figure 5.7: The CP$_{\text{III}}$ iterative algorithm (with fixed β, η, and $|C|$).

In case both above modifications take place, i.e. F^{CP1} is replaced by $F^{CP1'}$ and $F^{CP*1'}$ is replaced by $F^{CP*1'}$, a new algorithm, the $\mathrm{CP_{III}}$ algorithm (Figure 5.7), is derived in much the same way the CP algorithm (Figure 5.5) is derived. This algorithm was introduced, grounded on a different information-theoretic motivation, by Marx, Dagan & Shamir, 2004.

It is possible to replace only one of the two terms: either F^{CP1} by $F^{CP1'}$, or F^{CP*1} by $F^{CP*1'}$. If only F^{CP1} is replaced by $F^{CP1'}$, we derive the $\mathrm{CP_{II}}$ algorithm, which consists of update steps CP1', CP2', CP3' of the $\mathrm{CP_{III}}$ algorithm and steps CP*1, CP*2 of the original CP algorithm. The $\mathrm{CP_{II}}$ algorithm was introduced with intuitive motivation by Dagan, Marx & Shamir, 2002. If only the F^{CP*1} term is replaced by $F^{CP*1'}$, we derive the $\mathrm{CP_{I}}$ algorithm, with iterative cycle consisting of update steps CP1, CP2 of the original CP algorithm and CP*1', CP*2', CP*3' of the $\mathrm{CP_{III}}$ algorithm.

5.4 Experimental Work

In order to examine the capabilities of the algorithmic framework described above, we have conducted experiments on both artificial and real-world (textual) data.

The method of IB and ID with side information (IB-SI and ID-SI, described in Subsection 5.2.3) suggests a seemingly sensible alternative to our approach to CP clustering. As we aim at obtaining clusters that are not correlated with the given pre-partition, our setting is naturally mapped to the side information setting by considering the pre-partition W as the additional set of irrelevant features Y^{-}. Adapting this convention, our experimental results include comparison with IB-SI and ID-SI results and thus also with the plain IB and ID algorithms, which are equivalent to their corresponding SI version when the parameter γ is set to 0.

5.4.1 Experiments with Synthetic Data

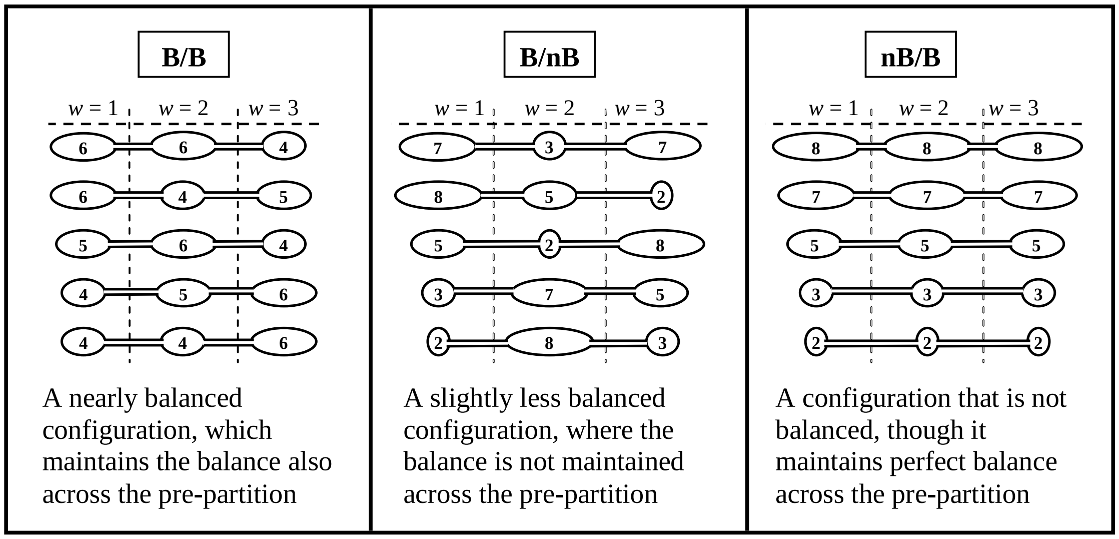
In general, the CP method is designed to tackle cases where each one of the pre-given subsets is relatively homogenous and might be characterized by salient subset-specific structure. The target of the CP method is to neutralize such within-subset homogeneities and regularities and to reveal structure that is persistent across the pre-partition part, even if it is not as salient on average. The following setting aims at assessing the level by which the CP method reveals hidden cross-partition structure in the presence of more salient clustering configuration, with clusters that do not cut across the given pre-partition but are rather restricted to elements of one of the pre-given subsets.

5.4.1.1 Setting

Our synthetic setting consisted of 75 virtual elements, pre-partitioned into three 25-element subsets, corresponding to three admissible values of the variable W (in our formalism, for each element x in the w-th subset, $w = 1$, 2, or 3, $p(w|x) = 1$). On top of this pre-partition, we tailored together two independent (exhaustive) clustering configurations. One of them – the target configuration – will capture cross-subset correspondences, while the other – a masking configuration – will represent within dataset structure. We would like to see if and how the CP method reveals the target configuration, even in cases where the masking configuration is considerably more salient.

1. The target *cross-W* clusters: five clusters, each with representatives from all three pre-given subsets. In different experiments, we used three distinct cross-partition configurations, differing in the level of global balance (equal vs. diverging cluster sizes) and cross-partition balance (equal vs. diverging sizes of cluster-subset intersections). Figure 5.8 provide the details of the three different cross partition configurations that were used.
2. The Masking *within-w* clusters: six clusters, each consisting of either 13 or 12 of the 25 elements of a particular subset with no representatives from the other subsets.

The same set of features was used to direct formation of clusters. However, each cluster, of both target and masking configurations, was characterized by a designated subset of the features. Associating an element with the cross-W cluster and with the within-w cluster to which it is assigned by construction was carried out by specifying a count of co-occurrences with each one of the features designated as characteristic to both clusters. The masking within-w clusters were systematically designed to be more salient than the target cross-W clusters. The within-w clusters had more designated features than the cross-W clusters, per cluster (60 vs. 48) and in total ($6 \times 60 = 360$ vs. $5 \times 48 = 240$). In addition, the simulated co-occurrence counts associating elements with their within-w cluster (a base level of 900) were higher than the co-occurrence counts associating elements with cross-W cluster (700 in a *salient CP configuration* setting, 400 in a *non-salient CP configuration* setting). Noise (a random positive integer < 200) was added to all counts associating elements with the designated features of their within-w and cross-W clusters, as well as to approximately one quarter of the zero counts associating elements with features designated for other clusters.



A nearly balanced configuration, which maintains the balance also across the pre-partition

A slightly less balanced configuration, where the balance is not maintained across the pre-partition

A configuration that is not balanced, though it maintains perfect balance across the pre-partition

Figure 5.8: The three different cross-partition clustering configurations used in the synthetic data experiments. The numbers of elements in the intersections of each one of the five CP clusters and each one of the three pre-given subsets are indicated.

5.4.1.2 Results

Performance level in the synthetic data experiments was measured relatively to the target cross-*W* configuration – one of B/B, B/nB and nB/B (see Figure 5.8) – that was used in constructing the particular data being tested. Each one of the three cross-*W* configurations underlay two different types of datasets, distinguished by the saliency levels of the target relatively to the masking configuration (400 or 700 target co-occurrence counts versus 900 masking co-occurrence counts; see previous subsection). This gives a total of six experimental settings. The variance between the different test cases within each experimental setting was the result of two random factors: the random noise added and the overlap, i.e., the number of shared elements, between within-*w* clusters and cross-*W* clusters (partition of elements to clusters was random hence cluster overlap was random too).

In each one of the six experiment settings, we tested six different methods – the four CP method variants (5.3.3, 5.3.4) and the two SI variants (5.2.3) – each over a range of values of the parameters γ in the SI algorithms, and η in the CP algorithms. The values of γ and η were kept fixed throughout each run, while the β parameter was gradually incremented in order to produce the target five clusters (see Subsection 5.3.3.2). For values outside the tested parameter ranges, the majority of runs did not end with five clusters. Reasons for not obtaining the target number of clusters were that the run did not converge after a large number of iterative cycles or, in the CP case, it could also converge to too few clusters (Subsection 5.3.3.2). Each one of the reported results was averaged over 200 runs, differing by the noise and by within-*w* and cross-*W* cluster overlap.

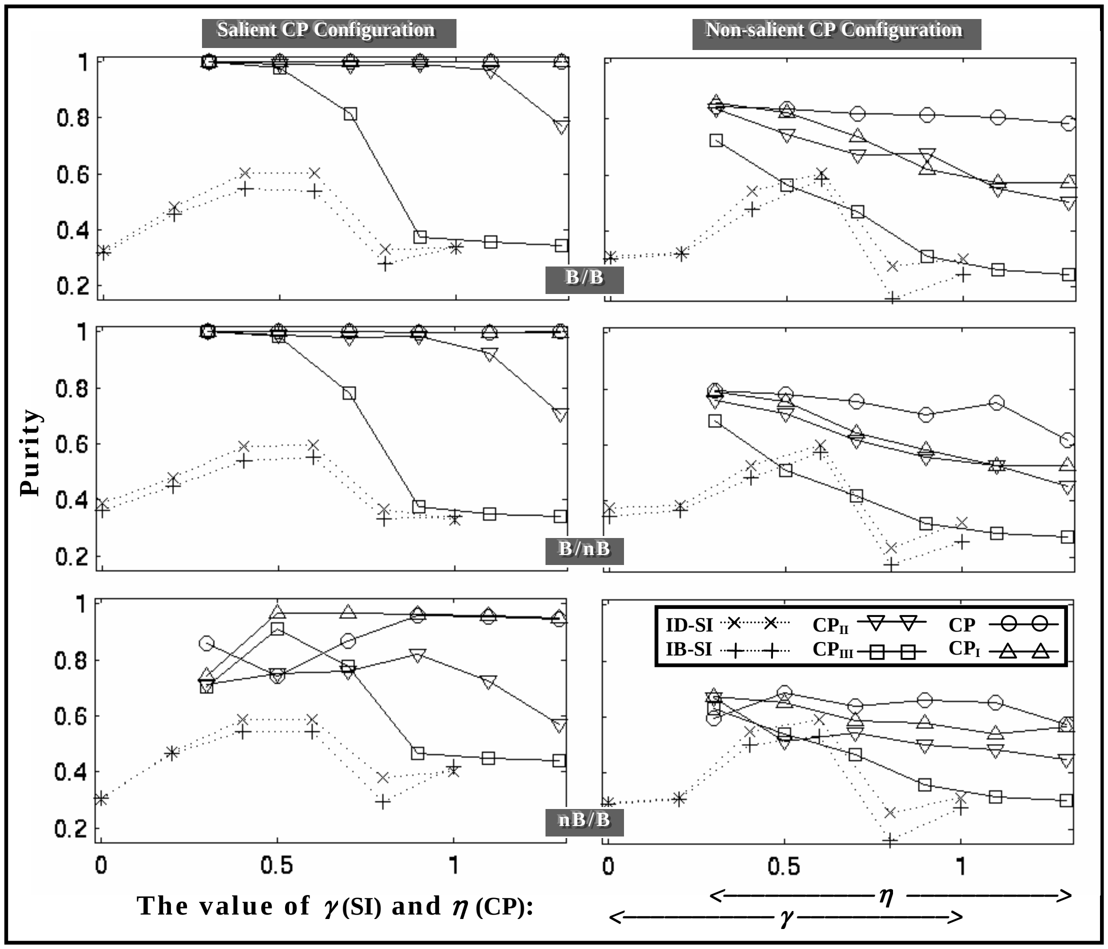


Figure 5.9: The results of the experiments with synthetic data by the six algorithms tested – CP, CP$_I$, CP$_{II}$, CP$_{III}$, ID-SI and IB-SI – in the six experimental settings. See the previous subsection for the description of the B/B, B/nB and nB/B cross-partition configurations and the difference between the salient and non-salient CP configuration settings.

As the target number of clusters was given by construction, we used the straightforward *purity* measure (the overall proportion of elements correctly assigned; Chapter 3, Subsection 3.3.1). Our measurements refer to the "hardened" version of the probabilistic output of the tested methods, i.e. the deterministic clustering configuration where each element *x* is considered a member in the cluster *c* with highest *p*(*c*|*x*).

Figure 5.9 displays purity results produced by the six algorithms tested in the six experimental settings. The graphs displayed indicate that the various versions of the CP approach perform better than the SI approach in the majority of cases examined. The difference is most notable in the easier tasks, i.e., in the more balanced configurations and especially in the "salient" setting, where some of the CP variations consistently achieve almost perfect reconstruction of the target configuration, over a large part of the tested η range. In addition, the SI algorithms tend to have a relatively narrow best-performance picks around certain γ value, while the CP performance is in general more stable across a large range of η values.

The "priorred" CP variations, especially CP_I and the CP_{III} which include the prior in their first step, tend to produce along with clusters that capture the target patterns small clusters that are often not part of the target configuration but rather seem to be the result of the added noise and interactions with the masking configuration. The plain CP algorithm was overall the best among the CP variations, while CP_{III} was the worst. The differences between the four versions are less noticeable in the lower η value range. Finally, there is a persistent advantage, though very small, to the ID-SI over the IB-SI. We will discuss possible reasons for the differences between the methods in the concluding section of this chapter.

5.4.1.3 Oscillatory Endless Loops

The previous subsection described the behavior of the CP algorithm (several variants) in cases where the data was drawn based on a prominent underlying cross partition structure and the algorithm converged in most cases. In the next section we will see that this nice behavior is indeed the case with the real world datasets we worked on. The current subsection shed some light on those cases where the underlying cross partition structure is not as prominent and consequently the algorithm is sometimes trapped in an endless loop.

We investigated this behavior experimentally through setting similar but simpler than the one described in Subsection 5.4.1.1 above. This simpler setting included eight virtual elements, pre-partitioned into two four-element subsets, with competing cross-*W* and within-*w* configurations. The two cross-*W* clusters included four elements each, two from each subset; the four within-*w* clusters, two within each subset, consisted of two elements each. The competing configurations were set to be in disagreement: the two elements of any within-*w* cluster were assigned to different cross-*W* clusters. Each cluster of both types was characterized by a single feature. As in the first setting, the within-*w* clusters were designed to be more salient with virtual element-feature co-occurrence count fixed on 100 (for the two elements of each cluster). The virtual cross-*W* co-occurrence counts varied in the different experiments between 0 and 100. Noise was added to all counts associating elements with the designated features of their within-*w* and cross-*W* clusters, as well as to approximately one quarter of the zero counts associating elements with features designated for other clusters.

On the eight-element dataset described above, we ran the (non-priorred) CP algorithm and applied the procedure of modifying the β value gradually in order to find the exact value inducing a split into two CP clusters. Table 5.1 shows the change in the proportion of times where the algorithm encountered an endless oscillatory loop as a function of the saliency of the cross-*W* clusters. The more salient the cross-*W* clusters are, less cases of an endless loop are observed. These results are based on 200 runs for each tested value of cross-*W* count. In order to decide on an endless loop we counted 500,000 iterations (convergence whenever obtained typically occurred in tens or hundreds of iterations and never more than a few thousands). The η value was fixed on 1.0 in these experiments.

Table 5.1: The proportion of endless loop cases decreases as the relative weight of the Cross-*W* element-feature count increases. The Cross-*W* element-feature counts in this table are weighed comparatively to a fixed "within-*w* feature count" of 100.

Cross-*W* element-feature avg. count	0–55	60	65	70	75–85	90–100
Proportion of endless loop cases	40-44%	25%	9%	5%	2%	1%

As Table 5.1 shows, in this simple setting when the relative weight of the features associated with CP clusters is low, the CP algorithm is trapped in a loop in 40% or more of the cases. Table 5.2 brings one such example where the weight of the cross-*W* relative weight is fixed on zero (so its corresponding features are not present at all). In this example, the CP algorithm oscillated for β values between 1.858 and 2.738. For lower β values no split occurred in the data. In this example, but not always, two clusters were produced for β values higher than 2.738. Note that this behavior and parameter values can change due to exact values of initial assignments, split initialization, etc.

Table 5.2: An example for a setting where the CP algorithm does not converge (for particular β and η values). Each line in the table contains the co-occurrence count information for another one of the eight data elements. The underlying structure, as reflected by the feature counts, includes four within-subset clusters of two elements each, plus "noise" associating elements with clusters to which they are not assigned by construction.

Data Elements:		Features associated with …			
		Within-A cluster 1	Within-A cluster 2	Within-B cluster 1	Within-B cluster 2
Subset A	**A1**	10	—	—	—
	A2	9	1	—	—
	A3	1	9	—	—
	A4	—	8	—	2
Subset B	**B1**	—	—	10	—
	B2	—	2	8	—
	B3	—	—	—	10
	B4	1	—	1	8

5.4.2 Application to Religion Data

For testing our method on real world data, we used the religion-related datasets and the evaluation method (Jaccard coefficient scores) that were used in the previous chapter and were described in detail in the experimental part of the previous chapter (Chapter 4, Subsection 4.4.2.1; see also Appendix A). We note that the CP method, in difference from the coupled clustering method, can be used for identifying correspondences across more than two religions at a time, as demonstrated in the following subsection.

5.4.2.1 Results

We survey below some of the CP clustering output, exemplifying it through results produced by the plain CP algorithm with $\eta = 0.48$, applied to all five religions together ($|W| = 5$). We have found that even the most coarse two-cluster partition generated by the above method is highly informative and illuminating. It reveals two major aspects that seem to be equally fundamental in the religion domain, which we termed “spiritual” and “establishment” aspect. The cluster that corresponds to the “spiritual” aspect of religion incorporates terms related with theology, underlying concepts and religion-related personal experience. Many of the terms assigned to this cluster with highest probability, such as *heaven*, *hell*, *soul*, *god* and *existence*, are in common use of several religions, but there are religion-specific words such as *atman*, *liberation* and *rebirth*, which are key concepts of Hinduism. The “establishment” cluster contains names of schools, sects, clergical positions and other terms related with religious establishment, geographical locations and so on. Keywords assigned to this cluster with high probability are mainly religion specific: *protestant*, *vatican*, *university*, *council* in Christianity; *conservative*, *reconstructionism*, *sephardim*, *ashkenazim* in Judaism and so on (there are few keywords that are common to several religions, for instance *east* and *west*).

The same two-theme partition consistently repeats also when the CP method is applied to pairs and triplets of religion. As far as our corpora represent faithfully the domain and our method extracts well the relevant information, these two factors can be considered the two universal constituents upon which the very notion of religion is laid.

CLUSTER 1 “Schools”
Buddhism: america asia japan west east korea india china tibet
Christianity: orthodox protestant catholic west orthodoxy organization rome council america
Hinduism: west christian religious civilization buddhism aryan social founder shaiva
Islam: africa asia west east sunni shiah christian country civilization philosophy
Judaism: reform conservative reconstructionism zionism orthodox america europe sephardim ashkenazim

CLUSTER 2 “Divinity”
Buddhism: god brahma
Christianity: holy-spirit jesus-christ god father savior jesus baptize salvation reign
Hinduism: god brahma
Islam: god allah peace messenger jesus worship believing tawhid command
Judaism: god hashem bless commandment abraham

CLUSTER 3 “Religious Experience”
Buddhism: phenomenon perception consciousness human concentration mindfulness physical livelihood liberation
Christianity: moral human humanity spiritual relationship experience expression incarnation divinity
Hinduism: consciousness atman human existence liberation jnana purity sense moksha
Islam: spiritual human physical moral consciousness humanity exist justice life
Judaism: spiritual human existence physical expression humanity experience moral connect

CLUSTER 4 “Writings”
Buddhism: pali-canon sanskrit sutra pitaka english translate chapter abhidhamma book
Christianity: chapter hebrew translate greek new-testament book text old-testament luke
Hinduism: rigveda gita sanskrit upanishad sutra smriti brahma-sutra scripture mahabharata
Islam: chapter surah bible write translate hadith book language scripture
Judaism: tanakh scripture mishnah book oral talmud bible write letter

CLUSTER 5 “Festivals and Rite”
Buddhism: full-moon celebration stupa ceremony sakya abbot ajahn robe retreat
Christianity: easter tabernacle christmas sunday sabbath jerusalem pentecost city season
Hinduism: puja ganesh festival ceremony durga rama pilgrimage rite temple
Islam: kaabah id ramadan friday id-al-fitr haj mecah mosque salah
Judaism: sukoth festival shavuot temple passover jerusalem rosh-hashanah temple-mount rosh-hodesh

CLUSTER 6 “Sin, Suffering and Material Existence”
Buddhism: lamentation water grief kill eat hell animal death heaven
Christianity: fire punishment eat water animal lost hell perish lamb
Hinduism: animal heaven earth death water kill demon birth sun
Islam: water animal hell punishment paradise food pain sin earth
Judaism: animal water eat kosher sin heaven death food forbid

CLUSTER 7 “Community and Family”
Buddhism: child friend son people family question learn hear teacher
Christianity: friend family mother boy question woman problem learn child
Hinduism: child question son mother family learn people teacher teach
Islam: sister husband wife child family marriage mother woman brother
Judaism: child marriage wife mother father women question family people

Figure 5.10: A sample from a seven-cluster output CP configuration of the religion data: the first members – up to nine – of highest $p(c|x)$ within each religion in each cluster. Cluster titles were assigned by the author. See appendix E for the full configuration.

Partitions into clusters of finer granularity still seem to capture fundamental, though more focused, ingredients of religion. The partition into seven clusters reveals the following topics (our titles): "schools", "divinity", "religious experience", "writings", "festivals and rite", "material existence, sin, and suffering" and "community and family". The relation between this seven-cluster configuration to the coarser two-cluster configuration can be explained in soft-hierarchy terms: the "schools" cluster and, to some lesser extent "festivals" and "family", are related with the "establishment" aspect reflected in the partition to two, while "divinity", "religious experience" and "suffering" are clearly associated with the "spiritual" aspect of religion. The remaining topic, "writings", is equally associated with both. The probabilistic framework enabled the CP method to cope with these composite relationships between the coarse partition and the finer one. Figure 5.10 details the first members – up to nine – of highest $p(c|x)$ within each religion in each of the seven clusters. The whole two- and seven-cluster configurations produced by the CP method, including $p(c|x)$ and $p(c)$ values, are given in Appendix E.

It is interesting to have a notion of those features y with high $p^*(c|y)$ (Subsection 5.3.2.3). Many of those features are in fact identical with some of the corresponding cluster's terms, especially ones that are common to several religions but, occasionally, also ones that are specific to one religion but are mentioned in discussions regarding other religions. We exemplify those typical features, for each one of the seven clusters, through four of the highest $p^*(c|y)$ features that did *not* have a dual role of clustered keywords (more comprehensive lists are brought in Appendix E.):

- "schools" cluster: *central*, *dominant*, *mainstream*, *affiliate*;
- "divinity" cluster: *omnipotent*, *almighty*, *mercy*, *infinite*;
- "religious experience" cluster: *intrinsic*, *mental*, *realm*, *mature*;
- "writings" cluster: *commentary*, *manuscript*, *dictionary*, *grammar*;
- "festivals and rite" cluster: *annual*, *funeral*, *rebuild*, *feast*;
- "material existence, sin, and suffering" cluster: *vegetable*, *insect*, *penalty*, *quench*;
- "community and family" cluster: *parent*, *nursing*, *spouse*, *elderly*.

The above terms were not initially pre-marked but rather the CP clustering approach, through its feature-cluster re-association mechanism, has pointed each such feature as particularly informative with regard to the cluster with which it is associated.

The topic-based perspective on the religion domain, as obtained from the demonstrative results above, can be related with works in the field of the comparative study of religion. One notable source for drawing such relation is Ninian Smart's work, for instance his book *dimensions of the sacred* (1996).

Smart specifies six different dimensions spanning those essential aspects, in the light of which world religions can be understood and compared. These dimensions are the *ritual* dimension, the *mythic or narrative* dimension, the *experiential and emotional* dimension, the *ethical and legal* dimension, the *social* dimension and the *material* dimension. In addition, Smart separately mentions *political effects* of religion. It is obvious that this analysis is not geared by keyword counts, but leans on what appears to be abstract and deep considerations and knowledge. However, these dimensions fit rather nicely to the partition to "spiritual" versus "establishment" aspects suggested by the two-cluster partition of the keyword data produced by the CP method. Specifically, the *mythic or narrative* and the *experiential and emotional* dimensions are related with the "spiritual" aspect, while the other dimensions, including political effects have to do with the "establishment" aspect. In addition, some relations to our seven-cluster based topics can be observed. Two dimensions that are unambiguously mapped onto our clusters are the *ritual* dimension, which is mapped to the "festivals and rite" cluster, and the *experiential and emotional* dimension, which is mapped to the "religious experience" cluster. More associations, though less obvious, exist such as the ones relating the *mythic or narrative* dimension with the "writings" cluster and the *social* dimension with the "community and family" cluster.

Another example of a theoretical view on religion that is related with our empirical outcome is Kedar Nath Tiwari's book *Comparative Religion* (1992). This book systematically reviews several religions including the five religions we refer to, each religion in a separate chapter. Subsection titles are identical in all chapters. Thus, the repeating titles give a notion of what the author considers as the main factors common to all religions. The subsection titles are specified as follows (we indicate the ones that are unambiguously mapped to one of our seven clusters): *god* (mapped to our "divinity" cluster), *world*, *man*, *evil and suffering* (mapped to "material existence, sin, and suffering" cluster), *life after death*, *human destiny*, *discipline* and *sects* (mapped to "schools" cluster).

To summarize viewing our results in light of related studies of comparative religion, our findings cannot be said to capture the details of any particular theory. From the two examples above, we see that we can not expect such theories to largely overlap with one another. Interesting partial mapping between the clusters generated by the CP method and ingredients of existing theoretical views nevertheless exist and worth mentioning. In the next subsection, we relate our results further with knowledge from religion studies, this time more systematically and in quantitative terms.

5.4.2.2 Quantification of the Overlap with the Expert Data

We quantitatively evaluated results of the cross-partition clustering method applied to the religion data. Results by three different versions of the CP algorithm, produced with different fixed η values, were examined. As baselines, we used the basic Information Bottleneck (IB) method applied to the union of the subsets, Information Bottleneck with Side Information (IB-SI) and our coupled clustering method (Chapter 4).

As in the previous chapter, we compared our results to classes of terms manually constructed by experts of comparative study of religion (as described in detail in Chapter 4, Subsection 4.4.2.3; see also Appendix C). The same 17 test cases involving pairs of religions were examined: all ten datasets made of pairs of subsets corresponding to all possible ten religion pairs were compared to the classes contributed by expert I. Four out of the same ten religion pairs could be further compared to classes contributed by expert II and three of the ten could be compared to the classes by expert III. Here as well, keywords not used by the expert were eliminated from the evaluated output *after* the completion of the clustering process. Again, we quantify the agreement between the output resulting from applying the CP method to a pair of religions at a time and the classes provided by the experts in terms of Jaccard coefficient (see Chapter 3, Subsection 3.3.2).

Given that the CP method produces probabilistic "soft" clustering, we had the option of using the "soft" Jaccard score variant. However, the scores produced by the soft version were similar to the standard Jaccard scores obtained from a "hardened" configuration (i.e., the "soft" scores did not reflect the potential added value of identifying real multi-assignments or ambiguities). We therefore used the standard version applied to the hard configuration resulting from assigning each element x to the cluster c with highest $p(c|x)$, but the expert data was considered probabilistic in cases of multi-assignment (as explained in more detail in Chapter 3, Subsection 3.3.2.1) similarly to the way the IB method was evaluated (as a baseline) in the previous chapter.

One further aspect regarding Jaccard scores, which is independent of the hard versus soft issue discussed above, refers to the adaptation of the scores to the cross-partition clustering setting. In the previous chapter, we used a version specifically adapted to coupled-clustering. This version counts only cross-subset pairs, while discarding the within-subset pairs altogether in a manner resembling the actual calculations conducted by the coupled clustering method, which relies solely on between-subset similarity values. Since the cross-partition approach of this chapter is essentially centroid-based and, as such, can be viewed as oriented towards clusters as wholes rather than towards the cross-subset associations set by the output configuration (Chapter 3, Subsection 3.3.2.1), we found it appropriate to apply here the standard version, which counts both within-subset and cross-subset pairs.[6]

[6] Without specifying the detailed scores, we denote that the Jaccard coefficient version adapted to coupled clustering produces in general higher scores in evaluating the CP method results, but the difference is not as noticeable as it is for the coupled clustering case described in Chapter 4.

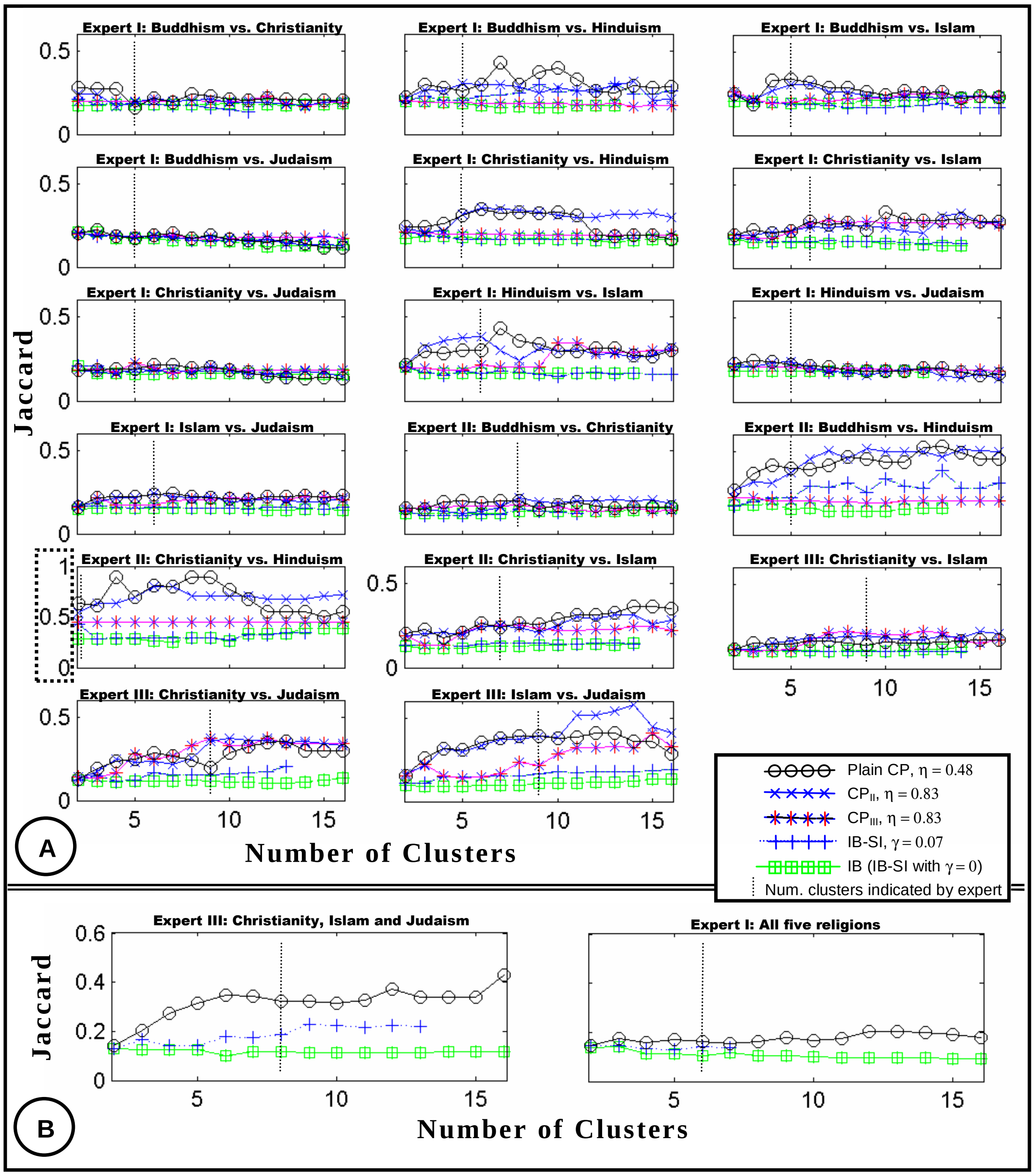


Figure 5.11: The religion keyword cross-partition clustering results evaluated relatively to the expert classes. Jaccard scores are shown for cluster numbers range from two to 16, for all 17 cases: ten by expert I, four by expert II and three by expert III. The algorithms in use: different versions of the CP clustering method with different η values (the plain version with $\eta = 0.48$, CP$_{\text{II}}$ with $\eta = 0.83$ and CP$_{\text{III}}$ with $\eta = 0.83$), the Information Bottleneck method and the Information Bottleneck with Side Information (with $\gamma = 0.07$). Note the different scale used for the "Expert II: Christianity vs. Hinduism" case, marked by a dotted frame.

The results of the experiments from all 17 test cases are displayed in Figure 5.11.A. The variations on the CP algorithm are demonstrated through three of the different versions: CP (with $\eta = 0.48$), CP_{II} (with $\eta = 0.83$) and CP_{III} (with $\eta = 0.83$). The other methods represented in Figure 5.11 are Information Bottleneck method and the Information Bottleneck with Side Information (with $\gamma = 0.07$). As already demonstrated, the CP method can be applied to data pre-divided into more than two subsets. Hence, apart from the 17 pairwise test cases, we also tailored a triple Christianity-Islam-Judaism keyword classification, based on the pairwise cross-religion comparisons provided by expert III. Similarly, we tailored also a configuration of term classes involving all five religions, based on the contribution of expert I. These configurations were used for evaluating the performance of the CP method on data pre-partition into three and five subsets. The results are displayed in Figure 5.11.B. As in the coupled clustering case (Chapter 4), the numbers of clusters indicated by each expert, denoted by dotted vertical lines in Figure 5.11, do not predict the actual number of clusters in the highest scored configuration. Thus, the target number of clusters was not assumed known, so that a whole range of output configurations of two to 16 clusters is scored. Figure 5.11 demonstrates that those CP versions using no prior or prior of one kind – CP (with $\eta = 0.48$) and CP_{II} (with $\eta = 0.83$) – perform better than the CP_{III} version (with relatively high $\eta = 0.83$) using both priors. All CP versions, however, perform better than the IB and IB-SI methods.

As discussed before (Section 5.3.3.2), parameter values outside a certain range prevent some of the examined algorithms from converging to sufficiently many clusters, or direct convergence to smaller number of clusters than the desired number. In our experiments, we used parameter values that allowed the formation of 16 clusters for all datasets. The existence of such parameter values is not obvious, as the datasets involving different pairs of religions differ from one another to much higher extent than the synthetic datasets (Subsection 5.4.1). However, it was not hard to find η values that worked well for all religion datasets.

Table 5.3 specifies, separately for each of several examined methods – CP, IB/IB-SI, and coupled clustering with the multiplicative cost function (Chapter 4) – an average over the 17 mean-values obtained by averaging over the range of examined numbers of clusters. As the table shows, the various η values that we tried yielded results that were similar on average, with the exception of slightly deteriorated performance by the CP_{III} version with the higher η value (which is in apparent agreement with the results of the synthetic experiments, Section 5.4.1). In contrast, the highest γ value that worked reasonably well for the IB-SI experiments, 0.07, was not sufficient for producing the desired number of clusters in the five religion case (note that this value is far lower than the optimal values in the synthetic experiments).

Table 5.3: Averages, over all 17 religion pair comparison cases, of means of 2–16 cluster Jaccard scores, recorded for the four CP method versions, each with four different η values.

	$\eta = 0.48$	$\eta = 0.56$	$\eta = 0.67$	$\eta = 0.83$
CP	0.2789	0.2778	0.2829	0.2816
CP_I	0. 2700	0.2716	0.2854	0.2954
CP_{II}	0.2701	0.2727	0.2820	0.2779
CP_{III}	0. 2664	0.2733	0.2656	0.2241
SI ($\gamma = 0.07$)	0.1812			
CC (multiplicative)	0.1806			
IB	0.1634			

Table 5.4: Average Jaccard scores over the 17 religion comparison evaluation cases. Each case is represented by the mean value (and, in parentheses, the best value) of all examined number of clusters. In parentheses: the average over the best score of each case. In the lower part of the table, difference values that were not found statistically significant (two-tailed *t*-test with 16 degrees of freedom, significance level 0.05) are marked with an asterisk.

Algorithm	Jaccard Score
Means ± standard deviations **of the 17 scores averaged over (**best of**) all examined numbers of clusters 2–16**	
CP ($\eta = 0.48$)	**0.2789**±0.1283 (0.3540±0.1692)
CP_{II} ($\eta = 0.83$)	**0.2779**±0.1319 (0.3452±0.1579)
CP_{III} ($\eta = 0.83$)	**0.2241**±0.0676 (0.2651±0.0809)
IB-SI ($\gamma = 0.07$)	**0.1812**±0.0525 (0.2214±0.0804)
CC (multiplicative)	**0.1806**±0.0514 (0.2475±0.0725)
IB	**0.1634**±0.0472 (0.1889±0.0601)
Means ± standard deviations **of the 17 coupled differences between scores averaged over (**best of**) 2–16 clusters**	
CP – CP_{II}	**0.0009**±0.02298* (0.0088±0.0610*)
CP – CP_{III}	**0.0548**±0.0793 (0.0889±0.1282)
CP_{II} – CP_{III}	**0.0538**±0.0835 (0.0801±0.1055)
CP_{III} – IB-SI	**0.0429**±0.0583 (0.0437±0.0949*)
CP_{III} – CC	**0.0435**±0.0518 (0.0176±0.0858*)
IB-SI – CC	**0.0006**±0.0393* (–0.0261±0.0657*)
CC – IB	**0.0172**±0.0489* (0.0586±0.0861)
IB-SI – IB	**0.0177**±0.0342 (0.0325±0.0528)

Along with the same averages over 17 mean-values, Table 5.4 specifies (in parentheses) averages over 17 scores, each of which is the best of all examined numbers of clusters in a test case. In addition to the five methods exemplified in Figure 5.11, the table incorporates results of the coupled clustering (CC) method (Chapter 4) with the multiplicative cost function. The lower part of Table 5.4 confirms, based on the same data, the statistical significance of the differences between the CP versions and the IB, IB-SI and CC methods, which were recorded already in Figure 5.11 and Table 5.3.

5.4.2.3 Agreement between the Experts

Agreement between each two experts that contributed evaluation data for the same pair of religions is quantified through measuring the overlap between the classifications provided by the two experts, based on the commonly used terms. Together, there was a total of 16 cross-expert evaluation cases involving religion pairs: there was one religion pair (Christianity-Islam) to which all the three experts generated evaluation data and additional five religion pairs for each of which evaluation data was provided by two of the three experts. The average of the Jacccard scores quantifying agreement in the 16 cases is specified in the first line of Table 5.5 (same as in Chapter 4). In order to have common grounds for comparison with the common-term-based agreement between experts, terms not used by *either* expert were discarded also from the evaluated clusters (after the clusters were formed) leaving in the evaluated clusters only the terms used by both experts.

Table 5.5: The cross-expert agreement Jaccard score, along with the coupled differences of this score from means over of the 16 cross-expert religion pair evaluation cases (means over 2–16 cluster Jaccard scores, see text). The methods examined are CP, CP_{III}, SI and CC. The difference between the expert agreement and the plain CP method is marked with an asterisk to denote it is not statistically significant (two-tailed *t*-test with 16 degrees of freedom, significance level 0.05) in contrast to the other differences recorded.

	Jaccard Score
Means ± standard deviations **for cross-expert agreement scores**	
Cross-expert Agreement	**0.467**±0.2246
Means ± standard deviations **of the 17 coupled differences from expert agreements averaged over all 2–16 clusters**	
Expert agreement – CP (η = 0.48)	**0.405** (**0.0620**±0.1403*)
Expert agreement – CP_{III} (η = 0.83)	**0.293** (**0.1741**±0.1552)
Expert agreement – SI (γ= 0.07)	**0.201** (**0.2657**±0.2087)
Expert agreement – CC (multiplicative)	**0.202** (**0.2651**±0.2630)

Table 5.5 compares the agreement between experts to the clustering produced by the various methods examined, evaluated based on terms common to the two experts between which agreement is measured. It is not surprising that the Jaccard scores obtained based on those relatively few "consensual" terms are considerably better than the results in the previous subsection.

Table 5.5 explicates evidence that, on average, the results produced by the plain CP and the CP_{II} methods score closely to the cross expert agreement, up to a level where the difference is not statistically significant. Particularly, the CP and CP_{II} scores are noticeably closer to the expert agreement score than then the score of any of the other methods, including IB-SI and the CC methods and the CP_{III} variation as well.

5.5 Discussion

In this chapter, we have introduced and demonstrated the cross partition clustering method. In order to address the cross-partition clustering task, this method follows the regularities of the feature distribution in the data, in much the same manner as done by familiar standard probabilistic clustering methods, such as IB and ID methods (reviewed extensively above in Section 5.2). In difference from the standard clustering techniques, the CP method considers an additional source of information, namely a pre-partition of the data. It turns that the target regularities in the feature distribution – those cutting across the subsets of the given pre-partition – are not straightforwardly distinguishable from the subset-specific information that the CP method seeks to neutralize. Providing the means for distinguishing the cross-subset part of feature information from the subset-specific part is a key innovative aspect of the cross-partition method.

The initial motivation to developing the CP method was studying the notion of analogy (see Chapter 2, Section 2.2), with which it copes from a novel perspective. Each subset of the given pre-partition of the data represents one of the several systems between which analogy is drawn. Our method stresses those attributes that are common to the analogized systems within a framework similar to that of standard feature-based data clustering, which also realizes additional constraints related to the target of identifying a correspondence between the systems.

The maximum entropy principle plays in the CP method a key role in interleaving the principles that direct the CP task within one iterative loop. The iterative loop of the CP algorithm is divided to two parts, each taking care of one of two principles: forming clusters based on the relevance feature distribution and ensuring that formation of clusters is carried out independently of the pre-partition to subsets. Accordingly, the maximum entropy principle is also applied twice. The fact that the iterative loop of the cross partition algorithm realizes separately, through different update steps, two different directions has to do with our inability to specify a cost function that is minimized by the cross partition algorithm as can be done for many other related methods (such as the information distortion, information bottleneck and information bottleneck with side information methods; all reviewed above Subsections 5.2.1, 5.2.2 and 5.2.3).

As mentioned, the core idea of the CP method's algorithmic mechanism lies in the step implementing feature-cluster re-association (Subsection 5.3.2.3). The associations of the characterizing features with the formed clusters are biased so that these associations become *independent* of the given pre-partition of the data.

The IB-SI approach (Chechik & Tishby, 2003; Subsection 5.2.3), which we consider a natural alternative to our method, can be understood in similar terms. In the way we implemented the IB-SI method (5.4), with the set of pre-given subset labels (our W variable) taken as the set of "negative features" (the IB-SI's Y^- variable), the IB-SI method aims at overall de-correlation of the formed clusters C from the information regarding the given pre-partition. As Y^+ and Y^- are correlated to some extent (otherwise there is initially no problem), such global treatment to C implies that de-correlating the unwanted C-Y^- association seems to affect undesirably the wanted C-Y^+ association and vice verse. In distinction, the CP method de-correlates the *associations* between features and the formed clusters, i.e., the detailed C-Y joint distribution, from the pre-partition. That way, the relation between C and Y is selectively focused on those regularities that cut across W, which fits our target more accurately as verified by the empirical results.

The IB-SI method, like the ID and IB methods (Section 5.2) and in distinction from the CP method, incorporates all its underlying considerations, including the target de-correlation between the set of "negative features" and the formed clusters as discussed above, within a single cost term (Eq. 5.13). This seems to be advantageous from the point of view of clarifying what the method aims at. The behavior of the CP method, namely convergence onto a steady state involving several equations is more complex and less intuitive to describe. The CP method, however, consistently outperforms the IB-SI and the other tested methods in the cross-partition clustering experiments we have conducted. This empirical superiority might suggest, for instance, a potential utility in expressing each one of the considerations underlying a composite task through a different term and seeking a solution that mutually constrains all terms relatively to one another rather than optimizes an all-encompassing cost term. Of course, such direction is yet to be formulated and examined in general terms – the CP method only exemplifies this option.

One further aspect in the comparison between the IB-SI to the CP approach is that both iterative algorithms are not guaranteed to converge (particularly, the iterative IB-SI algorithm is not guaranteed to minimize the IB-SI cost term). Nevertheless, the CP iterative algorithm have shown an empirical advantage throughout our experiments in being more tolerant to changes in its parameter values, while the IB-SI requires tuning its parameter within a more restricted range for optimal performance.

The CP method improves significantly also relatively to the coupled clustering method introduced in the previous chapter. The coupled clustering method is a heuristics that is bound to some oversimplifications, most notably the assumption of a given similarity measure and the restriction to utilize only between subset similarities. The CP method not only utilizes the data more directly and thoroughly, but it does so in a more principled manner based on considerations of maximizing relevant information and the maximum entropy principle. Comparing the empirical results of the two chapters, we notice the difference in the Jaccard scores resulting from the two methods. Further, also the demonstrative examples show, to the best of our judgment, that while the CC results give an idea regarding a seemingly random selection of themes that are part of the religion domain, the CP outcome provides much more of an exhaustive and balanced sub-topical picture of the whole domain. The CP method reveals meaningful constituents that, to our understanding, indeed can be considered as the main building blocks of religion along various resolution levels.

We conclude this chapter with a remark regarding the priorred versus the non-priorred versions of the CP algorithm (Subsection 5.3.4). Including priors allow the formation of clusters of more diverging sizes, while the lack of priors poses a bias towards the limit of uniform distribution of elements over the clusters. On one hand, it can be argued that in some well-motivated information theoretic sense the methods that apply prior reveal what is “really” in the data. On the other hand, allowing small clusters along with large ones gives rise also to small clusters that are the result of “noise” rather than “true” information. Our conclusion is that it might be worth to include priors in cases where there is a reason to believe that the process is going to capture the “true” underlying structure very accurately. If this is the case, the utility of using a prior is intensified as far the (“true”) distribution of elements over the clusters is from uniform distribution. It seems, however, that in many real data clustering applications high purity level cannot be granted. We believe that the superiority of the non-priorred version of the CP algorithm in our religion comparison task demonstrates well the widespread case where it is better not to apply any type of prior. Also in the synthetic experiments the non-priorred CP algorithm lost its superiority in settings of *both* imbalanced configuration and relatively low level of noise (see bottom left hand side of Figure 5.9).

Chapter 6:
Discussion and Further Directions

In this work, we have defined and studied a new unsupervised computational learning task: automated identification of correspondences across a dataset pre-divided into several – two or more – subsets. We have developed methods that accomplish this task and we have demonstrated them on synthetic data as well as on real world data extracted from un-annotated texts. As our methods are introduced using general formulation that does not depend on their application to texts, our approach is potentially utilizable for a wide variety of real-world problems. At the same time, it opens new perspectives on the analogy making task, which has been typically associated with cognitive concepts and mental processes such as discovery and creation.

Our work extends the data clustering task, in a way that enables coping with analogy or correspondence identification. Correspondences are identified by way of assigning together corresponding elements from different subsets into the same cluster. As we have emphasized (Section 3.1), the straightforward application of a standard clustering technique would not address well the above task. This is particularly true when each of the pre-given subsets given as input is relatively homogenous and overall not very similar to the other subsets. In such cases, a standard clustering method would tend to produce clusters with elements restricted to a single subset. Our results, however, demonstrate the capability of modified clustering techniques to reveal correspondences between the subsets as required, rather than subset-specific themes.

As mentioned (Subsection 2.1.1.2), the data clustering problem is formally ill posed. In practice, the quality of a proposed solution for a specific is assessed in terms of the requirements of the specific application. Our modified data-clustering problem is subject to the same type of ambiguity and, in fact, the potential source of ambiguity is even heaped on. While ambiguity in the original data clustering task results from the lack of a definite criterion for how elements should be grouped together, the extended task adds on top of that a potential ambiguity regarding constructing the matches across two or more given subsets. In spite of being formally ill posed, the data clustering task has been studied extensively. We hope our new task would be recognized as a useful tool that deserves further study just as the standard data-clustering notion on which it is based.

We have developed two different data-clustering based computational methods that identify correspondence across several given subsets of data elements. The first of which, *coupled clustering* (Chapter 4), is based on a recent cost-based pairwise clustering framework. In this setting, the distributional data representation that is typically given needs conversion to pairwise similarities. The second method, *cross partition clustering* (Chapter 5), extends clustering methods grounded on information theoretic accounts and is directly based on co-occurrence distribution. Each of these two methods bases its strategy on few principles, pertaining to the essence of the task of identifying correspondences.

There are several advantages to starting from studying a setting based on deterministic ("hard") data-clustering, as we have done in Chapter 4, rather than the probabilistic or "soft" variations. Deterministic clustering is technically and conceptually simpler and it constructs more definite and easily interpretable clustering configurations. The coupled clustering method restricts the similarity values generally considered by pairwise clustering methods to similarities of elements that are not in the same subset ("between-subset similarities"). It is thus guided by the working assumption that information within a subset is not supposed to impact directly the correspondences formed across subsets but, rather, the information resulting from comparing two subsets, i.e., similarities between members of the different subsets, should be the factor to consider in constructing such correspondences. This assumption motivates the main original mechanism underlying the coupled clustering method: the cost-function that we have proposed (H^3, Eq. 4.14), which incorporates two complementary principles. The first one is the underlying principle of pairwise clustering in general:

A cluster should contain elements that are similar to one another.

The second principle turned to be the underlying idea of the coupled clustering method:

In order to be formed, a cluster must exceed some level of prominence in both subsets (as opposed to a cluster that is overall more prominent, but most of its members are concentrated in one of the subsets).

This second direction is realized through a geometric-mean term that is used for calculating average similarity in each cluster.

In summary, the coupled clustering method is a rather straightforward elaboration on the standard cost-based pairwise clustering setting, which only restricts the collection of similarity values under consideration to the collection of between-subset similarities. The essential drawback of the coupled clustering method might lie in its presumed working assumption. It is probable that similarities between elements of distinct subsets are more important for the emerged correspondence, but the policy of restricting the attention to these similarities is, in retrospective, just a preliminary rough direction. The coupled clustering method does not accommodate studying further this axiomatic assumption, so the questions of whether and to what extent the non-considered within-subset similarities are utilizable for the task of constructing context-dependent correspondences remain open. In addition, the intermediate stage of calculating pairwise similarity implies yet another source for loss of information, which could have contributed to the revealed correspondences.

The cross partition data clustering method, introduced in Chapter 5, extends coupled clustering along several aspects: it enables the identification of correspondences across more than two pre-divided subsets, and it produces probabilistic rather than deterministic clustering output. It also saves the intermediating similarity calculation stage, as it relies on vectorial (probabilistic) representation of the data, which is the original format of the data in many cases. Like coupled clustering, the cross-partition clustering method follows two guiding principles. The first of which is the underlying idea of probabilistic centroid-based clustering (Subsection 2.1.4.6):

Clusters are formed around feature-distribution based centroids.

(The centroids are averaged over individual distributions of members in proportion to their membership probability, and thus expected to approximate the feature distribution for the cluster elements). The other principle is the main novel idea in the cross-partition method:

The formed associations between the clusters and the feature distributions characterizing them should maintain independence from the given pre-partition to subsets.

In order to illustrate the kind of impact this principle has on outcome resulting from the first principle (that is, standard probabilistic clustering), assume that some features distinguish well between groups of elements within one of several pre-determined subsets, while having no discriminative value within other subsets. Such features are *not* expected to direct the formation of cross-partition clusters, as they would push toward clusters made of elements restricted to one subset. Rather, features that push towards inclusion of members from all subsets in some cluster are expected to guide the formation of clusters, even if overall they are not as salient.

This direction is analogous to the second principle guiding the coupled clustering cost term. Both give rise to a geometrical mean term. In the coupled clustering case, the scheme involving geometrical mean has been justified by the intuitive direction of keeping both cluster parts of a considerable size. However, a restrictive working assumption such as the one taken by the coupled clustering method (restricting attention to between subset similarities) is not present in the cross partition framework. As a rough equivalent to our coupled clustering restriction, we mention the use of the maximum entropy principle, which is applied to highlight the constraints posed by the two principles above. Rather than posing some initial guess regarding where to look for the desired information, the maximum entropy principle enforces the assumption that *nothing* is known beyond constraints derived from the stated guiding principles.

There are several technical matters in the cross partition framework that need to be studied further. For example: clarifying the role of the external parameters (η and β) and their interplay and understanding how the number of pre-given subsets ($|W|$) – especially when this number is large – affects the behavior of the algorithm.

An optional direction, where the CP method can be practically applied is identifying repeating themes, or "roles", in topical news articles (which is directly related to the task of template induction for *information extraction*). We did a preliminary investigation in this direction at earlier phases of the research (Marx, Dagan & Shamir, 2002). The news articles that we examined were focused on the topic of terrorist attacks. In this domain, the target roles – the organization that carried out an attack, the location, the weapon used and so on – are typically addressed by different terms in each article. We applied a method that clustered together terms associated with each different role. Thus, each of the generated clusters reveals a correspondence across the given articles, which may underlie a slot in an information extraction template.

A related direction currently being implemented is extending the cross-partition framework to semi-supervised learning. As we indicated (Subsection 2.1.5.3), several recent works proposed to constrain (or "to seed") data clustering, e.g., by pre-specifying lists of element pairs that must, or must not, share a cluster. This idea can be adapted to the information distortion and information bottleneck methods, as well: if assignment probabilities of some data elements are pre-specified (or constrained in other ways), a straightforward modification of the algorithm would minimize a cost term just like the original methods do[1]. By the same token, also the cross partition framework can enable pre-specifying, or constraining, assignments of some of the elements. In an ongoing project, we study a setting where the assignments of all elements of one of the pre-given subsets are known, so this subset forms a training set, while elements of the other subset are assigned to clusters as in the original CP method. The idea mentioned above of applying the maximum entropy principle to highlight these additional constraints through separate iterative update steps is incorporated as well in the same project.

[1] This can be verified with slight modification of lemmas 5.1 and 5.3; specifically, the sums in Eqs. (5.8) and (5.11) should be modified to reflect the constrained assignments.

In this work, we have approached a task related with abstract cognitive functions – construction of correspondences and analogies – through a simple extension of the elementary data-clustering setting. We see our success in coping with a seemingly complicated task by means of a relatively simple setting an appealing aspect of our work. There are additional unsupervised tasks, however, which aim at constructs more complex than standard clustering, for example Bayesian nets, or graphical models. As mentioned, the information bottleneck method described in Section 5.2 has already been generalized to producing more complex types of constructs (*multivariate information bottleneck*; Friedman et al., 2002). Extending the cross partition clustering method along the same direction would form a natural and interesting continuation of the current work, which might lead to more insights regarding analogy making and related tasks.

This work provides an original perspective on the study of analogies. Analogy making is one of those slippery tasks with no consensual pre-given definition or characteristics, but very central to intelligence and creativity. Each one of the existing approaches to analogy making indicates different aspects of analogy as the essential ones (as exemplified in Section 2.2, and discussed a bit further below). In fact, there is a deep disagreement with regard to what are the considerations that underlie analogies in practice (see for example Ch. 4 in Hofstadter et al., 1995). Suggesting a computational framework applicable to this notion, as we have attempted to do here, has been a fascinating challenge, even though it is clear that such suggestion is not going to achieve consensus among researchers in the field.

Our approach to analogy making relies on word co-occurrence distribution in the given data, rather than on hand-written rules or pre-coded data representation of the type used by some previous studies. This approach thus bridges between cognitive motivations and observations regarding analogy making and the familiar vector model, which has been extensively used for practical tasks such as similarity assessment and classification. The data clustering task on which our methods elaborate can be seen as a basic "cognitive" tool for concept discovery. Our work takes this general tool and adapts it to discovery of concepts that form an analogy or other non-trivial context-dependent correspondence.

The setting underlying our computational approach is considerably different than previous views of analogy making. As noted above, the two methods that we introduced ascribe the correspondence being formed to interplay between two different principles. In coupled clustering, these are shared pairwise similarity (across subsets) and simultaneous prominence of the formed cluster in both subsets. In the cross partition method, the underlying factors are communal feature distribution patterns and their independence on the pre-partition variable. To the best of our knowledge, the cross partition framework is the first to characterize correspondence formed across several subsets in terms of statistical independence.

Previous works have considered other issues as central to analogy making. Analogy is ordinarily conceived as a means for problem solving ("analogical reasoning"), an aspect on which we have not focused here. The mapping of relational structure is a crucial conception at the core of the structure mapping theory (Gentner, 1983), but our method does not elaborate on this aspect as well. Further developments of our framework might aim at capturing and emphasizing relational structure.

Several other works also considered the retrieval problem: identifying the optimal object, which would allow the construction of an analogy to a given target object (Forbus, Gentner and Law, 1995). Our work does not address this point, as well: we examine two or more given element subsets, without accounting for how these subsets, or the systems they represent, have been chosen at the first place. The approach that we have introduced, however, is formulated in a general enough manner to allow the incorporation of aspects such as the above ones. For instance, in order to compare the quality of several candidate analogies, we might use cost-related criteria (in coupled clustering), or examine the dynamics of the algorithm (in cross partition clustering; e.g., assessing the quality of candidates according to the β value required for producing a fixed number of clusters).

There is a notable aspect that has been raised by other authors, particularly Hofstadter et al. (1995), which our approach seems to address in some sense: the emergent and fluid nature of the formed solution, which has to do with mental processes of creation and discovery. In some resemblance to the Copycat program (Subsection 2.2.2), our clustering mechanisms are based on aggregation of local changes that gradually evolve to a global solution of the problem at hand, while a temperature variable gradually introduces a more deterministic configuration. Further, our approach allows the formed solution to depend greatly on the context. When a particular subset is matched with different subsets, different themes might be revealed and mapped onto one another, in distinction, for example, from an approach that first clusters each subset independently and then map the independently clustered subsets onto one another. In this respect, as well, our approach accords with Hofstadter et al.'s view regarding the context dependent nature of analogies.

The computational mechanisms that we employ are essentially simpler than the ones suggested by Hofstadter et al. (refer to Subsection 2.2.2.2) and hence they are more liable to inspection and analysis. Hofstadter et al. advocate restricting the scope of investigation to artificial toy problems, allowing "looking at a problem together with its 'hallo' of variant problems" (Hofstadter et al., 1995). We have started with an approach that is inherently simpler. Thus, our approach might capture the analogy making problem only partially (though having potential to incorporate more aspects later on).

Particularly, our methods at their current stage might lack some subtleties addressed within the Copycat program. On the other hand, the principled computational machinery that we suggest allows cleaner demonstration of the impact of systematic manipulations on the input (see description of our experiments with synthetic data, Sections 4.3 and 5.4.1). Yet, without getting to the complex issue of how to evaluate and compare analogies, we think that our method captures something of their emergent and fluid nature. And above all, the most notable advantage we recognize in comparison to previous methods pertaining to analogy making is the immediate applicability of our methods to real-world problems and data.

In this work we have demonstrated our approach mainly on textual data. With no prior specialization or training in the study of religions, our program was able to identify analogous factors shared by several religions in varying levels of resolution: “spiritual” versus “establishment” dimensions in a coarse view and aspects such as “sacred writings”, “rite and festivals” and “sin and suffering” in a more detailed level. These findings are in apparent agreement with previous specialized comparative religion studies that are based on a systematic comparable approach. For the purpose of systematic evaluation, we have measured the overlap between our outcome and religion-related term clusters provided by experts and found their match very close to the level of agreement between experts.

Co-occurrence data and, more generally, data in vectorial representation are very common in many fields: artificial vision, biology, psychology and competitive intelligence, to mention just a few. As the formulation of the methods introduced in this work does not depend on any specific application, we hope they will be applied in the future to a large variety of problems and domains.

Appendix A: Religion Data

A.1 The Sub-corpora

	Buddhism	Christianity	Hinduism	Islam	Judaism
Original corpus size	1.44 (8.66)	1.89 (10.77)	2.14 (12.99)	1.51 (8.72)	1.56 (9.18)
Lemmatized corpus	0.76 (5.57)	0.98 (6.71)	1.20 (8.63)	0.77 (5.38)	0.83 (5.87)
size in millions of word tokens (megabytes)					
# documents	58	44	44	52	44
Encyclopedic entries	8	8	8	8	8
FAQ files	4	4	4	2	5
Online periodicals	4	4	1	6	5
Other web-sites / online books	42	28	31	36	26

- The sub-corpora used for this work contain the sub-corpora used by Marx, Dagan, Buhmann and Shamir (2002), extending them by 25-50%.
- Inclusive documents have been chosen: ones that essentially refer to a religion as a whole, rather than being pre-limited to a specific aspect or school. This way, we have tried to create sub-corpora of general character that repeatedly refer to a variety of aspects on roughly the same level of detail. This makes the practice of filtering out key-terms appearing on less than four documents somewhat more reliable.
- Some of the documents have been created as a merge of several pages appearing on the same Internet site. For example, each "online-periodical" document consists of up to tenth of individual articles.
- We have made efforts to exclude texts that literally repeat over different web pages.
- The use of a POS tagger and lemmatizer is also new relatively to Marx et al. (2002). There, filtering of function words was controlled by a pre-determined list. Here, we have identified content words by their part of speech, leaving only the lemmatized nouns (including names), verbs, adjectives and adverbs. Numbers tagged as cardinal or ordinal numbers were replaced by ~*card*~ and ~*ord*~ signs
- Some of the most prevalent alternative part-of-speech tags have been attached to the lemmatized word tokens. For example, *mean/V* stands for occurrences of the lemma 'mean' tagged as a verb, while *mean* stands for the noun sense, which is more prevalent within our sub-corpora, as well as for any other part of speech that the tagger has (erroneously) attached to the same lemma. Other tags are */N* for names or nouns and */J* for adjectives and adverbs. The alternative tags are attached whenever the alternative repeats 50 times or more (in all sub-corpora).

A.2 The Features

For each specific cross religion comparison, the features used – i.e., counted content words co-located with the clustered key-terms – are those occurring in both compared sub-corpora (at least twice in each corpus). The numbers of features used for each comparison:

Common to all 5 religions:	6796	Christianity, Islam and Judaism:	7768
Buddhism and Christianity:	8717	Buddhism and Hinduism:	9905
Buddhism and Islam:	7973	Buddhism and Judaism:	8735
Christianity and Hinduism:	10410	Christianity and Islam:	8604
Christianity and Judaism:	9641	Hinduism and Islam:	9438
Hinduism and Judaism:	10454	Islam and Judaism:	8563

Following is the list of the 58 features that are among the 100 most common features in at least four of the five sub-corpora. The numbers in brackets indicate the number of joint occurrences in which the feature is involved (in order to calculate *p*(*y*) for any feature *y* within the configuration incorporating all sub-corpora, one should divide the number in brackets by the total count of co-occurrences in all sub-corpora):

have (99303), **not** (71697), **god** (54503), **do** (42993), **one** (36608), **say** (30779),
life (25722), **man** (20262), **other** (20243), **people** (20219), **make** (19733), **only** (19065),
give (18919), **come** (17978), **world** (17579), **so** (17402), **time** (17059), **also** (17020),
being (16685), **~card~** (15591), **many** (15433), **as** (15258), **way** (15077), **know** (15029),
more (14830), **see** (13531), **then** (13507), **word** (13203), **day** (13160), **go** (13115),
first (13014), **take** (12690), **most** (12570), **become** (12336), **good** (12277), **even** (12273),
great (12037), **believe** (11307), **human** (10786), **call** (10656), **out** (10527), **year** (10076),
live (9990), **find** (9937), **own** (9877), **such** (9575), **use/V** (9260), **book** (8928),
very (8889), **up** (8880), **two** (8875), **person** (8411), **mean/V** (8386), **now** (8271),
state (8194), **same** (8149), **bring** (8135), **teach** (8050).

The total co-occurrence count in the corpus (all sub-corpora): **4892150**

The rest of the 100 most common features within each individual corpus follow. The numbers in brackets indicate the co-occurrence count within the individual corpus. In order to calculate feature probability conditioned on the corpus $p(y|X_i)$ for any feature *y* one would divide the bracketed number by the total number of co-occurrences within the corpus, which appear at the end of each list.

Buddhism

buddhist (8452), **buddha** (7601), **buddhism** (6470), **practice** (3344), **teaching** (3236),
mind (3036), **meditation** (2307), **monk** (2068), **path** (1949), **suffering** (1869),
tradition (1866), **thing** (1774), **right** (1733), **truth** (1717), **develop** (1678),
just (1622), **religious** (1620), **death** (1586), **action** (1569), **understand** (1538),
religion (1490), **enlightenment** (1468), **sense** (1467), **follow** (1448), **teacher** (1439),
lead (1430), **nature** (1426), **order** (1421), **think** (1385), **three** (1372), **other/N** (1357),
term (1352), **text** (1343), **spiritual** (1295), **experience** (1284), **form** (1269),
different (1214), **part** (1202), **arise** (1200), **existence** (1199), **sangha** (1174),
well (1168), **school** (1165), **four** (1163).

The total co-occurrence count in the corpus: **812850**

Christianity

jesus (15446), **christ** (14275), **church** (7144), **lord** (6303), **sin** (5438), **son** (5041),
holy (4918), **bible** (4260), **faith** (4226), **thing** (3833), **christian** (3740), **father** (3693),
christian/J (3251), **save** (3131), **gospel** (3127), **death** (2993), **heaven** (2954),
tell (2803), **speak** (2803), **power** (2700), **work** (2655), **just** (2609), **paul** (2539),
heart (2513), **prayer** (2472), **john** (2430), **salvation** (2415), **write** (2393), **name** (2339),
get (2334), **baptism** (2277), **think** (2268), **ever** (2240), **body** (2232), **receive** (2211),
love (2151), **law** (2144), **grace** (2144), **child** (2143), **therefore** (2078), **holy/J** (2067),
scripture (2062), **testament** (2035), **want** (2021), **die** (2009).

The total co-occurrence count in the corpus: **1281079**

Hinduism

hindu (10614), **india** (5554), **hinduism** (4262), **religion** (3975), **temple** (3529),
religious (2854), **spiritual** (2835), **indian** (2748), **yoga** (2672), **worship** (2612),
lord (2535), **child** (2387), **soul** (2204), **ancient** (2160), **family** (2097), **culture** (2067),
vedic (1943), **sri** (1864), **body** (1835), **mind** (1774), **include** (1726), **part** (1725),
form (1689), **school** (1681), **system** (1644), **swami** (1627), **krishna** (1625),
philosophy (1617), **tradition** (1607), **knowledge** (1590), **different** (1586),
dharma (1581), **nature** (1534), **vedas** (1519), **karma** (1502), **practice** (1480),
sacred (1475), **new** (1473), **place** (1452), **ritual** (1449), **today** (1447), **scripture** (1423),
high (1420).

The total co-occurrence count in the corpus: **1002100**

Islam

allah (12391), **prophet** (8795), **islam** (7799), **muhammad** (4130), **muslims** (3980),
messenger (3254), **islamic** (3097), **religion** (2994), **follow** (2663), **muslim** (2483),
law (2207), **woman** (2132), **belief** (2079), **worship** (2078), **faith** (2058), **reveal** (1975),
prayer (1973), **muslim/N** (1965), **ask** (1903), **knowledge** (1844), **peace** (1843),
verse (1842), **heart** (1749), **holy** (1685), **true** (1658), **revelation** (1612), **jesus** (1587),
order (1573), **accord** (1570), **create** (1484), **name** (1442), **qur~an** (1429), **fact** (1406),
accept (1367), **fast** (1359), **send** (1357), **thing** (1333), **lord** (1333), **message** (1322),
truth (1319), **right/N** (1300), **place** (1293), **last** (1292), **believer** (1287), **well** (1275).

The total co-occurrence count in the corpus: **913241**

Judaism

jewish (10726), **torah** (6557), **rabbi** (4150), **judaism** (3666), **jews** (3541), **law** (3482),
israel (3456), **woman** (2562), **jew** (2328), **moses** (1940), **child** (1935), **name** (1845),
prayer (1793), **community** (1738), **temple** (1723), **religious** (1706), **write** (1687),
commandment (1604), **create** (1585), **part** (1569), **tell** (1514), **land** (1511), **begin** (1507),
just (1438), **include** (1421), **place** (1396), **talmud** (1352), **spiritual** (1351),
speak (1350), **synagogue** (1348), **tradition** (1336), **new** (1330), **father** (1313),
reform (1305), **movement** (1275), **service** (1259), **accord** (1238), **abraham** (1238),
well (1206), **rabbinical** (1206), **understand** (1199), **jerusalem** (1192).

The total co-occurrence count in the corpus: **881569**

A.3 The Clustered Keyword Sets

The sizes of the keyword sets are as follows:

Buddhism – 227; Christianity – 235; Hinduism – 177; Islam – 221; Judaism – 232.

The whole sets can be seen in the results Appendix E. Here, the clustered keyword sets are exemplified through arbitrarily chosen ~10% subsets (we have picked elements number 1, 11, 21, … and so on, according to their lexicographic order). Along with each element, we indicate its three most prominent features that are not included in any of the above lists of common 100 features per corpus. The individual feature lists below demonstrate well the information conveyed by the thousands of features that are not very frequent. Each feature is preceded by its relative rank in terms of number of co-occurrences with the particular element. The bracketed numbers, that is the count of joint occurrences, divided by the total co-occurrence count of the element *x*, gives the respective conditional probabilities $p(y|x)$. The total number of *x* co-occurrences divided by the total number of co-occurrences within the corpus (or within all sub-corpora), which has been indicated previously, gives the probability $p(x|X_i)$ (or $p(x)$).

Buddhism

Key-term	Features (co-occurrence count)			Total count
Abbot	2.monastery (13)	4.here (7)	7.there (5)	461
Asceticism	4.five (33)	8.wander (20)	13.austerity (16)	2307
Being	6.sentient (200)	13.living (122)	35.happiness (77)	24953
Burma	1.thailand (36)	2.lanka (35)	5.cambodia (15)	795
conditioning	3.process (9)	4.consciousness (8)	5.condition (7)	544
Discipline	2.rule (31)	6.monastic (27)	7.code (21)	2683
Emptiness	3.phenomenon (25)	5.realize (22)	8.realization (20)	2132
Family	6.friend (30)	10.leave (24)	11.member (20)	3239
full~moon	1.month (26)	3.moon (15)	5.night (12)	505
Hinduism	15.caste (13)	23.principle (9)	24.orthodox/J (9)	2194
Karma	5.result (91)	6.bad (79)	8.rebirth (71)	7730
Liberation	8.achieve (20)	9.insight (20)	12.attain (19)	2754
Meet	10.group (17)	13.need (16)	28.attend (12)	3041
noble~truths	6.suffer (47)	8.noble (44)	10.cause (36)	2937
Phenomenon	1.mental (34)	3.emptiness (29)	4.physical (27)	2472
psychologist	2.modern (7)	4.philosopher (4)	5.view (3)	311
Robe	1.wear (35)	4.bowl (18)	5.yellow (14)	1297
Sanskrit	1.pali (44)	5.language (24)	9.translate (17)	1616
Society	5.individual (32)	7.social (28)	14.member (20)	4698
Student	4.western (18)	10.master (13)	14.zen (11)	1806
Text	4.pali (67)	6.early (55)	14.group (36)	7680
Universe	8.phenomenon (19)	16.everything (16)	18.entire (15)	2566
Wisdom	2.compassion (100)	5.perfection (63)	12.virtue (38)	6313

Christianity

Key-term	Features (co-occurrence count)			Total count
Abraham	2.promise (91)	4.seed (50)	10.isaac (32)	2866
Association	4.unitarian (9)	7.evangelical (9)	8.universalist (8)	668
Believing	32.baptize (75)	43.ye (60)	50.reason (55)	19264
Cardinal	2.bishop (13)	3.pope (10)	4.elect (9)	285
Confess	11.forgive (30)	19.mouth (22)	27.faithful (15)	3468
Doctrinal	8.trinity (67)	21.christianity (39)	26.principle (34)	8561
Evangelical	2.theology (67)	19.century (22)	22.group (19)	5851
Fire	1.lake (68)	4.hell (48)	6.burn (39)	3234
Gift	18.tongue (37)	20.christmas (35)	21.prophecy (33)	5455
Heaven	4.earth (268)	14.kingdom (100)	18.hell (84)	12142
Idolatry	4.forme (4)	12.note (3)	14.inordinate (3)	335
Jew	8.gentile (40)	12.christianity (31)	17.roman (23)	5074
Law	9.keep (72)	24.divine/J (39)	25.break (38)	9470
Mary	3.virgin/N (40)	6.mother (33)	7.joseph (28)	2248
Moral	5.goal (38)	8.evil/N (33)	10.acceptable (25)	4218
Passage	11.refer (29)	12.read (25)	14.meaning (22)	3843
Pope	5.ii (39)	6.bishop (33)	7.roman (25)	2273
Question	6.answer/V (118)	7.answer (106)	10.raise (45)	8434
Revelation	16.chapter (19)	22.special (16)	37.divine/J (12)	3480
Salvation	21.plan (52)	28.eternal (43)	32.necessary (37)	9720
Soul	8.win (64)	20.winner (45)	25.immortal (36)	7894
Teach	11.pray (56)	12.doctrine (56)	31.disciple (26)	7404
Trinity	2.doctrine (66)	16.incarnation (17)	20.unity (12)	2160
Worship	10.music (47)	24.sunday (30)	27.praise (28)	7227

Hinduism

Key-term	Features (co-occurrence count)			Total count
Advaitha	8.theory (5)	13.pure (4)	15.doctrine (4)	419
Attain	4.liberation (51)	12.eternal (32)	13.bliss (32)	4421
brahma~sutra	1.commentary (21)	2.upanishads (15)	3.gita (11)	293
classical	1.music (85)	3.dance (58)	9.sanskrit (16)	1942
divinity	9.mother (81)	17.self (59)	22.society (48)	12320
festival	4.celebrate (51)	7.annual (38)	9.hold (32)	3622
gita	2.upanishads (49)	10.commentary (23)	17.sutra (20)	3349
hymn	2.veda (41)	3.collection (30)	4.rig (29)	2263
karma	10.reincarnation (78)	11.bad (73)	13.past (62)	9117
mahabharata	1.epic (73)	3.gita (23)	7.puranas (18)	1788
philosophy	23.six (42)	35.science (34)	37.upanishads (33)	10430
question	3.answer (83)	4.answer/V (72)	13.scend (25)	4432
rigveda	2.hymn (46)	3.veda (35)	8.old (19)	1929
scholar	3.sanskrit (30)	4.western (30)	14.leader (15)	2707
smriti	4.manu (15)	5.remember (13)	6.literature (12)	784
student	5.university (55)	15.learn (29)	20.college (25)	5771
tradition	22.value (52)	33.art (44)	44.preserve (34)	14172
west	5.east (93)	12.astrology (63)	21.eastern (42)	11606

Islam

Key-term	Features (co-occurrence count)			Total count
abraham	7.noah (54)	10.ishmael (39)	12.adam (36)	3195
army	3.battle (16)	6.fight (10)	10.commander (9)	1648
bible	20.mention (7)	25.statement (7)	29.difference (6)	1306
christian	35.doctrine (24)	44.trinity (21)	46.missionary (21)	7241
creature	7.creator (28)	10.mercy (22)	11.universe (18)	1880
earth	16.creation (34)	20.face (26)	21.belong (26)	5823
faith	20.article (52)	26.deed (48)	44.reject (28)	9806
food	1.eat (57)	4.drink (33)	10.abstain (17)	2239
guide	12.mankind (54)	22.clear (36)	24.seek (35)	6521
house	5.enter (32)	7.build (28)	11.pilgrimage (24)	2838
ishmael	4.isaac (20)	7.jacob (17)	8.build (15)	854
kaabah	1.mecca (23)	4.build (17)	6.house (16)	996
marriage	6.wife (58)	8.divorce (42)	16.husband (26)	3801
mother	8.wife (30)	12.sister (26)	23.baby (15)	2758
pilgrimage	1.hajj (99)	2.mecca (77)	5.duty (38)	3113
purification	5.wealth (18)	23.alms (7)	24.intention (7)	1343
responsibility	11.hold (22)	12.social (21)	14.society (20)	3145
science	6.modern (34)	7.technology (33)	19.art (18)	4308
social	2.economic (78)	4.political (72)	8.society (49)	4999
submission	5.obedience (44)	9.total (24)	13.complete (17)	2005
testimony	6.confirm (12)	7.bear (12)	12.ramadan (8)	704
water	7.drink/V (21)	8.jug (19)	12.down (16)	2489
writing	12.read (5)	15.jail (4)	16.leaf (4)	685

Judaism

Key-term	Features (co-occurrence count)			Total count
abraham	2.isaac (92)	5.sarah (66)	6.jacob (52)	5074
ashkenazim	8.jewry (10)	12.germany (9)	15.custom (7)	1094
canaan	2.egypt (18)	7.israelite (7)	9.jacob (7)	674
command	12.keep (17)	14.love/V (16)	17.sanctify (16)	2980
discuss	4.issue (41)	14.debate (16)	15.chapter (16)	3951
europe	2.eastern/N (42)	5.western/N (17)	6.century (16)	1476
family	4.member (71)	5.friend (62)	14.home (31)	6527
gemara	2.commentary (14)	3.together (8)	7.answer (7)	620
hebrew	6.language (40)	11.union (29)	12.hebraic (28)	4374
humanity	11.creation (7)	14.history (6)	20.image (5)	1046
jesus	4.messiah (18)	15.consider (7)	19.individual (6)	1001
law	7.oral (126)	14.custom (75)	24.code (63)	17337
member	8.congregation (32)	12.committee (21)	13.group (18)	3391
mourn	2.period (36)	8.house (13)	12.destruction (9)	1088
people	27.choose (87)	38.egypt (73)	47.covenant (62)	26564
rabbi	8.ben (99)	10.congregation (98)	14.role (89)	21829
revelation	2.sinai (27)	6.creation (14)	8.divine/J (13)	1658
salvation	5.miracle (8)	9.redemption (6)	12.covenant (6)	776
service	7.healing (54)	9.morning (50)	16.attend (35)	6049
spirit	4.healing (35)	11.evil (16)	26.letter (9)	2535
talit	1.wear (34)	4.shawl (12)	5.corner (11)	714
text	6.biblical (46)	7.read (40)	9.meaning (31)	5107
wisdom	10.understanding (19)	15.solomon (15)	26.divine (10)	2555
zionism	10.secular (13)	12.political (12)	16.century (10)	1478

Appendix B:

Examples of Religion Coupled Clustering

Table B.1: Coupled clustering of Buddhism and Christianity keywords. Cluster labels were added by the authors. The 16th cluster of lowest average similarity is shown as well in this case.

Buddhism	Christianity
1. ***Institutional Organizations, Different Schools (Originating Places)*** (0.119308)	
asia china establish india japan religion religious sangha society tibet tradition	america catholic church establish evangelical jew rome tradition
2. ***Doctrine / Philosophy / Theology*** **A**	
begin cause connect consciousness discipline doctrinal element enlightenment ethic exist existence history moral phenomenon philosophy precept question rebirth social study west word zen	doctrinal history question religion religious theology
3. ***Sorrow / Suffering, Sin, Punishment and Reward***	
being death experience find human man meditation people sense suffer	being believing bible child death devil faith find god heaven jesus love man people sin soul suffer
4. ***Spiritual / Psychological States***	
anger attain attitude awaken awareness emotion family focus freedom impermanence karma liberation nirvana pain peace perception physical problem realm relationship root spiritual universe wisdom	experience human moral problem relationship spiritual
5. ***The main figures with some of their characteristics/companions***	
buddha dharma monk practice teaching	apostle baptism father gospel holy holy-spirit jesus-christ john paul prayer prophet salvation scripture spirit teach word worship write
6. ***The Written System / Scripture***	
book chapter literature pali-canon sanskrit scripture sutra text translate write	author book chapter greek hebrew language luke matthew new-testament old-testament passage refer translate writing
7. ***Elements of Narrative*** **A**	
country disciple house meet member monastery nun project retreat temple thai	city family jerusalem meet member ministry school service sunday
8. ***Elements of Narrative*** **B**	
child friend hear master son teacher	abraham angel baptize bless boy face faithful friend hear listen mother preach savior teacher voice woman
9. ***Elements of Narrative*** **C**	
animal eat forest kill robe sit tree water	animal blood bread earth eat eye fire fish home house pay water
10. ***Elements of Narrative D***	
birth king prince siddhartha-gautama	birth crucifixion disciple isaiah israel king kingdom lamb messiah minister moses perish predict reign win
11. ***Institutional Organizations, Different Schools and Traditions (Places)***	
america burma christian east found hinduism mahayana north role scholar school south sri-lanka Theravada	ancient baptist bishop council luther organization orthodox orthodoxy protestant role west
12. ***Doctrine / Philosophy / Theology*** **B**	
bodhisattva faith god law	authority command commandment confess divinity flesh forgiveness foundation gentile gift grant judgment justification law mankind mary miracle redemption resurrection revelation righteousness sabbath sacrifice saint sinner teaching
13. ***Doctrine / Philosophy / Theology*** **C**	
asceticism concentration eightfold-path generosity guide intention learn mindfulness noble speech spirit strength student teach training	believer guide learn peace repentance study

14. ***Doctrine / Philosophy / Theology*** **D**	
abhidhamma argue argument author canonical discourse discussed emptiness english five-precepts foundation meditator metaphysical noble-truths psychologist skilful tantra theory vehicle vinaya writer writing	argue argument ethic incarnation postmodern text theory trinity writer
15. ***Admired / Glorified Qualities***	
arhat behavior buddhahood buddha-nature deliverance deva dukkha gift grief heaven hell humanity karmic-law painful purity sacrifice salvation samsara sentient soul unwholesome	atonement condemnation earthly elect expression godly good-work hell humanity jeremiah love-of-god obey-god passion punishment reward sake sinful sinless violence
16. (poor match)	
abbot ajahn amida authority bhikkhu bodhi-tree branch celebration ceremony conditioning dalai-lama experiment founder full-moon hinayana instruction koan korea lama lamentation livelihood mandala mantra missionary ordain pilgrimage pitaka reincarnation sacred sakya samadhi story stupa trade vajrayana veda vietnam vipassana worship	adultery apostolic association bearing bethlehem cardinal christmas constantinople convert cross easter elijah eucharist found founder german good-friday idolatry instruction intercession john-the-baptist jordan-river lost mass monk no-other-god patriarch pentecost pilgrimage pope priestly reading-bible sacrament season soul-winning story student tabernacle thomas trade university vatican zion

Table B.2: Coupled clustering of Buddhism and Christianity keywords. Cluster labels were added by the authors. The 16th cluster of lowest average similarity is not shown.

Christianity	Hinduism
1. ***Doctrine / Philosophy / Theology basic principles***	
being believing death father god heaven holy jesus jesus-christ love man people sin spirit suffer word	being child god human man people soul
2. ***Doctrine / Philosophy / Theology A***	
history religion religious theology tradition	art bhakti caste civilization history language meditation philosophy question ritual sacred sanskrit science social south study teach theory tradition vedas
3. ***Institutional Organizations***	
bible book church evangelical find john write	ancient book find india shri temple west word write yoga
4. ***Spiritual / Psychological States***	
divinity experience human moral relationship spiritual	attain brahman consciousness discipline divinity experience freedom idea karma law purity sense shiva spirituality universe
5. ***Doctrine / Philosophy / Theology – Writings***	
apostle argument chapter confess foundation language prophet refer revelation scripture study	scripture teaching text
6. ***Doctrine / Philosophy / Theology***	
authority doctrinal establish faith gospel law paul question rome salvation teach teaching worship	christian dharma faith practice religion religious society
7. ***Tradition***	
america family home meet member ministry school service sunday	ashram ceremony country dance family festival foundation ganesh pilgrimage priest sadhu school student teacher
8. ***Elements of Narrative***	
animal blood child devil eat eye face fire soul water	animal death earth fire food kill spirit water
9. ***Sorrow / Suffering, Sin, Punishment and Reward***	
baptism believer birth command flesh forgiveness gift grant hell holy-spirit judgment kingdom moses pay peace problem punishment repentance resurrection reward righteousness sacrifice season sinner	birth heaven person sacrifice
10. ***Elements of Narrative***	
abraham baptize bless boy disciple faithful friend hear learn listen minister mother prayer preach saint savior voice win woman	devotee gift guru learn mother prayer son
11. ***Doctrine / Philosophy / Theology***	
adultery ancient apostolic argue commandment constantinople expression gentile guide instruction isaiah luther mary matthew messiah orthodoxy patriarch pope role sabbath teacher theory translate violence	authority doctrine upanishad
12. ***Institutional Organizations***	
Baptist bishop catholic convert council found german jew organization orthodox protestant university vatican west	aryan brahmin buddhism found founder jain muslim scholar shaiva
13. ***Scripture / Writings***	
author greek hebrew luke new-testament old-testament passage text thomas writer writing	author buddha classical epic gita hymn indus literature mahabharata poem poet purana ramayana rigveda story sutra writing
14. ***Elements of Narrative A***	
angel city earth house israel jerusalem king	demon hero holy indra king krishna rama sita star sun valley Vishnu
15. ***Doctrine / Philosophy / Theology***	
atonement condemnation crucifixion earthly ethic godly humanity incarnation justification love-of-god mankind predict redemption reign sake sinful trinity	atman existence humanity liberation manifestation moksha rebirth reincarnation samsara shakti

Table B.3: Coupled clustering of Buddhism and Christianity keywords. Cluster labels were added by the authors. The 16th cluster of lowest average similarity is not shown.

	Hinduism	Islam
1.	being find god india man people religion vedas	allah god mohamad people prophet quran religion word
2.	caste law religious social society	economy establish law moral practice problem responsibility social society
3.	animal art death divinity experience food human karma meditation practice sacred shiva soul spirit teach temple tradition west yoga	being find human life man
4.	ancient book language question sanskrit shri study text word write writing	arab book language religious scholar write
5.	dharma faith person	authority believer believing declaration deed enemy facing faith fast fight forbid jihad justice mankind messenger peace punishment question sin teach witness worship
6.	brahmin child devotee family kill mother son	brother child death father friend home husband marriage mother sister son wife woman
7.	gita guru scripture teaching	command companion guide hadith holy imam jesus learn moses revelation sheikh statement sunnah tawhid teaching
8.	author buddha classical dance epic foundation founder history idea literature mahabharata philosophy ramayana reincarnation scholar science shaiva spirituality theory upanishad yogi	civilization history philosophy science study tradition
9.	aryan buddhism christian civilization country found indus muslim south valley	africa arabia asia christian city country east empire india jew west
10.	ceremony darshan festival ganesh holy pilgrimage prayer priest puja rite ritual	haj house id kaabah mecah mosque pilgrimage prayer ramadan
11.	atman attain bhakti brahman consciousness discipline element existence liberation manifestation moksha purity rebirth samsara sense shakti universe	consciousness divinity exist existence physical spirit spiritual submission universe
12.	ahram jain learn school student teacher	bank company family meet school service student university
13.	birth earth fire gift heaven sacrifice sun vishnu water	animal bless creature earth food heaven hell paradise soul water
14.	authority doctrine freedom humanity	attitude calif charity code commandment doctrine female finance foundation freedom humanity judge judgment lawful master mission pain polygamy poverty preach purification race racial ruler share sufi testimony ummah wisdom
15.	agama brahma-sutra hymn poem purana ramanuja rigveda samkhya smriti sutra trimurti	author bible chapter dua fiqh hijrah noah pillars-of-faith ritual sacred scripture shariah shiah shii succession sunni surah translate writing

Table B.4: Coupled clustering of Buddhism and Christianity keywords. Cluster labels were added by the authors. The 16th cluster of lowest average similarity is not shown.

	Hinduism	Islam
1.	being find god india man people religion vedas	allah god mohamad people prophet quran religion word
2.	caste law religious social society	economy establish law moral practice problem responsibility social society
3.	animal art death divinity experience food human karma meditation practice sacred shiva soul spirit teach temple tradition west yoga	being find human life man
4.	ancient book language question sanskrit shri study text word write writing	arab book language religious scholar write
5.	dharma faith person	authority believer believing declaration deed enemy facing faith fast fight forbid jihad justice mankind messenger peace punishment question sin teach witness worship
6.	brahmin child devotee family kill mother son	brother child death father friend home husband marriage mother sister son wife woman
7.	gita guru scripture teaching	command companion guide hadith holy imam jesus learn moses revelation sheikh statement sunnah tawhid teaching
8.	author buddha classical dance epic foundation founder history idea literature mahabharata philosophy ramayana reincarnation scholar science shaiva spirituality theory upanishad yogi	civilization history philosophy science study tradition
9.	aryan buddhism christian civilization country found indus muslim south valley	africa arabia asia christian city country east empire india jew west
10.	ceremony darshan festival ganesh holy pilgrimage prayer priest puja rite ritual	haj house id kaabah mecah mosque pilgrimage prayer ramadan
11.	atman attain bhakti brahman consciousness discipline element existence liberation manifestation moksha purity rebirth samsara sense shakti universe	consciousness divinity exist existence physical spirit spiritual submission universe
12.	ahram jain learn school student teacher	bank company family meet school service student university
13.	birth earth fire gift heaven sacrifice sun vishnu water	animal bless creature earth food heaven hell paradise soul water
14.	authority doctrine freedom humanity	attitude calif charity code commandment doctrine female finance foundation freedom humanity judge judgment lawful master mission pain polygamy poverty preach purification race racial ruler share sufi testimony ummah wisdom
15.	agama brahma-sutra hymn poem purana ramanuja rigveda samkhya smriti sutra trimurti	author bible chapter dua fiqh hijrah noah pillars-of-faith ritual sacred scripture shariah shiah shii succession sunni surah translate writing

Table B.5: Coupled clustering of Islam and Judaism keywords. Cluster labels were added by the authors. The 16th cluster of lowest average similarity is not shown.

	Islam	Judaism
1.	allah being believing faith find god life man messenger mohamad people prayer prophet religion word	death find god israel law man people prayer rabbi torah women word
2.	book hadith question quran revelation scholar tradition write	bible book discuss hebrew history letter oral question talmud text tradition write
3.	authority economy law practice religious social society	authority community conservative establish intellectual member mitzvah orthodox rabbinical reform religion religious ritual role service social society status
4.	animal earth food forbid lawful water	animal eat food forbid kosher water
5.	creature divinity exist existence guide human mankind physical spirit spiritual universe wisdom	connect divinity exist existence experience expression human humanity physical relationship revelation soul spirit spiritual universe wisdom
6.	africa arab arabia asia christian civilization country east india jew west	america area ashkenazim center christian country east europe german north sephardim
7.	brother child daughter death family father friend home husband kill marriage mother sister wife woman	child family father hear home kill marriage mother son wife
8.	attitude charity code declaration facing fight foundation freedom humanity jihad justice meet moral problem responsibility share submission teach teaching ummah witness	faith freedom moral
9.	history learn philosophy school science student study sufi	ancient argue language learn liturgy mystic philosophy reconstructionism scholar school student study teacher teaching theology zionism
10.	city house kaabah madinah mecah mosque	city egypt exile house israelite jerusalem menorah priest scroll sea star synagogue temple temple-mount wear
11.	angel believer bless command deed enemy heaven hell judgment master pain paradise peace remember satan sin soul	angel bless command hashem heaven hide peace redemption reward sin teach
12.	fast friday haj id id-al-fitr pilgrimage ramadan salah surah	atonement calendar candle celebration festival holiday meal mourn passover purim read-torah rosh-hashanah rosh-hodesh sabbath sacrifice shavuot sukoth
13.	abraham abu-bakr ali companion holy imam jesus moses son	abraham chapter covenant david esther exodus holy isaac jacob joseph king moses mount-sinai prophet sarah
14.	army baghdad calif descendant empire establish israel istanbul jerusalem ruler succession tipu tribe	babylon canaan judah kingdom pharaoh rome tribe yhwh
15.	author bible chapter doctrine english fiqh judge language scripture shariah statement sunnah testimony translate writing	argument author code gemara halachah kabalah literary mishnah scripture siddur statement story tanakh writer writing zohar

Appendix C: The Expert Data

This appendix explicates the details regarding the data contributed by the experts for the religion comparison experiments in chapters 4 and 5. The first part contains a copy of the instructions that were provided to the experts. The second part includes the classes themselves.

C.1 Instructions for Participants

We run a research on fully automated detection of similar features within distinct but related domains. With your kind help, we would have the opportunity to validate our computational information-retrieval methods through a comparative investigation of religions.

Please follow the instructions below. Thank you very much for your contribution!

Zvika Marx, student in neural computation Ph.D. program

For the following five religions – Buddhism, Christianity, Hinduism, Islam, Judaism – you are requested to provide a simplified framework for comparing each two of them (Buddhism-Christianity, Buddhism-Hinduism, etc.) with one another.

For each one of the possible 10 pairwise comparisons, please write down in English:

(I) A list of titles for the main features and aspects that are similar (or resembling, or parallel, or equivalent, or analogous) in the two religions under examination.

(II) For each such a feature, for each one of the two religions under comparison, write down a list of key-terms that are associated with this feature.

Notes:

(1) **Additional Guidelines**:

(a) Please, try to address features that are commonly discussed in the literature and to use key-terms that are relatively wide spread. Please avoid the use of very rare terms as much as possible (even if they are much more to the point).

(b) Feature titles and key-terms may repeat (over different religion comparisons, repetition of the same key-term for both religions in the same feature etc.). However, keep in mind that we have an interest in exploring unique properties of each comparison.

(c) It would be much helpful if you could specify in brackets alternative spelling and strictly interchangeable terms to your key-terms, whenever such exist.

(d) The key-terms may definitely be names of people, places etc., but please try thinking also of key-terms that are not names, whenever such exist.

(2) Expected numbers of features and key-terms (rough, non-definite guidelines):

∗ Features –	minimum: about 5;	maximum: about 15	(for each pair)
∗ Key-terms –	minimum: 2-3;	maximum: about 10	(for each religion in each pair)

(3) This task is not meant to be laborious and the output is not expected to become an ultimate reference for comparative religion studies. It is more of getting a view on what you might have in mind now. Please do not consult the literature too extensively (unless you do it for your own reasons). Do concentrate on currently available knowledge. Whenever pointing out additional features and key-terms is not fluent, tend to the minimum values specified in (2) above.

(4) We would like very much to get your contribution also with respect to religions about which you feel less confidence, even if in such cases you are significantly less accurate and detailed. You are requested to fill in a self-estimated indication to your level of expertise for each comparison, so we could potentially monitor consequential effects if any.

(5) Although the list of all 10 pairwise religion comparisons is what we are after, partial contribution (resulting from lack of time, lack of knowledge, or any other reason) is still very much welcomed.

(6) Your name will not be associated with your specific contribution. Please let us know if you prefer not to be acknowledged at all.

(7) We might come back to you with a request to feedback in a later stage in our research. However, by participation in this stage you do not make any commitment to take part in the later stages.

(8) We would like very much to get comments and suggestions regarding your impressions of taking part in this procedure. We would be happy to explain and discuss further any aspect of our research.

The instructions above were followed by 10 pages, each containing a table such as the following, where the words **Religion1** and **Religion2** are respectively replaced by the following religion pairs:

(1) **Buddhism** and **Christianity**, (2) **Buddhism** and **Hinduism**, (3) **Buddhism** and **Islam**,

(4) **Buddhism** and **Jusaism**, (5) **Christianity** and **Hinduism**, (6) **Christianity** and **Islam**,

(7) **Christianity** and **Jusaism**, (8) **Hinduism** and **Islam**, (9) **Hinduism** and **Jusaism**

and (10) **Islam** and **Jusaism**,

1 **Features of Similarity**	**Religion1**	**Religion2**
	Key-terms Lists	

Level of confidence – please indicate a number between 1 to 5 _____________

1 – indicates minimal confidence (features and key-terms are based on very partial knowledge or even on mere intuition, I am not sure if many relevant aspects are covered, etc.)

5 – indicates maximal confidence (features and key-terms are based upon comprehensive knowledge, they seem to me as representing well the whole range of potential correspondences, etc.)

C.2 Religion-Related Term Classes Contributed by Experts

C.2.1 The Data Contributed by Expert I

Table C.1: We present the data contributed by this expert, as a cross-comparison involving all five religions. The expert provided the titles for the aspects of similarity, i.e. classes, and the list of religion pairs to which each similarity aspect is relevant. This data was conveyed to us in printed form, so we show the final version, after filtering out about 30% of the terms and further editorial changes, such as spelling modifications (the data from the other 2 experts, presented in the following sections, allows more detailed trace of filtered terms and changes that have been made).

Buddhism	Christianity	Hinduism	Islam	Judaism
Scriptures. (relevant for all religion pairwise comparisons)				
pali~canon koan mandala mantra sutra	new~testament old~testament apostle bible john luke matthew paul revelation	gita mahabharata upanishad vedas	hadith mohamad shariah sunnah	halachah mishnah siddur talmud tanakh torah
Beliefs and Ideas (relevant for all religion pairwise comparisons)				
buddha~nature noble~truths dharma dukkha emptiness nirvana reincarnation suffer tantra	jesus~christ love~of~god devil god cross fish heaven hell resurrection trinity	holy~people trimurti moksha atman brahman reincarnation	pillars~of~faith muhammad~the~prophet~of~allah tawhid allah heaven hell	judgment divine~creation mount~sinai no~other~god principles~of~faith temple~mount temple hell yarmulke exodus heaven menorah talit tefilin teshuvah
Ritual, Prayer and Holy Days (relevant for all religion pairwise comparisons, less Christianity–Hinduism)				
Gift meditation sacrifice stupa	eucharist baptism christmas confess sunday	kumbhamela festival meditation puja sacrifice	id~al~fitr charity fast friday haj kaabah mecah prayer ramadan	atonement read~torah circumcision passover prayer synagogue tefilin
Society and Politics (relevant for the following comparisons: Budd.–Judd., Chris.–Hind., Chris.–Islam., Hind.–Islam, Islam–Judd.)				
Dalai~lama bodhisattva lama monk	catholic church minister monk priestly protestant rome vatican	brahmin caste sadhu	calif ali imam shariah sheikh shiah sufi sunnah	hasid saint ashkenazim community david prophet rabbi sephardim Solomon synagogue
Establishments (relevant for the following comparison only: Christianity–Hinduism)				
monastery school temple	bishop cardinal church pope priestly	caste gift priest temple	imam mosque	high~priest temple priest levi mikveh rabbi synagogue yeshivah
Mysticism (relevant for all religion pairwise comparisons)				
meditation nirvana samadhi tantra	eucharist crucifixion love miracle saint suffer	ashram chakra darshan guru yoga	sufi	zohar ezekiel angel ari daniel kabalah malchut messiah sefirot shechinah
Learning and Religious Education (relevant for the following : Budd.–Chris., Budd.–Islam, Chris.–Islam, Hind.–Islam, Hind.–Judd.)				
meditation monastery monk sutra	divinity moral theology university	ashram guru	sheikh	gemara parsha talmud yeshivah
Names and Places (relevant for the following religion comparisons: Budd.–Hind., Chris.–Islam., Chris.–Judd., Islam–Judd.)				
siddhartha~gautama buddha	jesus~christ john~the~baptist jordan~river bethlehem jerusalem jesus john luke luther mary matthew paul rome thomas	varanasi arjuna brahma durga ganesh kali Krishna rama shakti shiva vishnu	jerusalem ali baghdad istanbul madinah mecah mohamad	isaac joseph jacob abraham adam jerusalem moses

C.2.2 The Data Contributed by Expert II

The second expert has provided the data presented in Tables C.2–C.5.

Table C.2: Buddhism–Christianity expert classes.

	Buddhism	**Christianity**
Life after death	Reincarnation, nirvana (nibbana) karma samsara	Resurrection heaven hell the day of judgement
Creatures that have special forces or abilities	Buddha Bodhisattva Deva	God Jesus Devil Angel
Cause and effect	The law of karma the cycle of reincarnation samsara	Sin – punishment ~~good act~~ - reward
Sacred places	Temple monastery ~~the places related to the life of the Buddha~~	Jerusalem Betlehem church ~~cathedral~~ ~~graveyard~~
Recommended acts	Dharma (dhamma) meditation ascetism ~~bhavana~~ ~~the way~~ (faith, strength, mindfulness, concentration, wisdom) ~~merit-making~~ pilgrimage	Praying reading Bible good works converting believing
Sacred times	Full-moon day ~~the day of the birth and enlightenment of the Buddha~~	Christmas Easter Pentacostal day Sunday
Special places for religious experts	Sangha monastery	~~Monastery~~ ~~convent~~ monk ~~nun~~
Sacred texts	Tripitaka ~~Kanjur~~ ~~Tanjur~~	The Bible New Testament Old Testament

Table C.3: Christianity–Hinduism expert classes.

	Christianity	**Hinduism**
Trimurti	Father Son and the Holy Ghost	Brahma Vishnu Shiva
sacred texts	The Bible Old Testament New Testament	Vedas Upanishads Mahabharata Brahmasutra

Table C.4: Buddhism–Hinduism expert classes.

	Buddhism	Hinduism
reincarnation	nirvana (nibbana) karma samsara	Moksha samsara karma
Meditation	Dharma (dhamma)	Dharma atman Brahman yoga
Ascetism	Sangha monastery pilgrimage	Pilgrimage meditation yoga
Guru	Buddha Bodhisattva Deva	Yogi ascetics brahminis mahatmas
Sacred texts	Tripitaka ~~Kanjur~~ ~~Tanjur~~	Vedas Upanishads Mahabharata Brahmasutra
~~*Syncretism*~~	~~Folk religion~~	Indus Veda Aryan ~~folk religion~~

Table C.5: Christianity–Islam expert classes.

	Christianity	Islam
Transcendent creatures	God, Jesus, angel, devil	Allah, Muhammed, angels devil, profets
Life after death	Resurrection heaven hell the day of the judgement	~~Predestination~~ paradise hell the day of the judgement
Holy scriptures	Bible	Qur'an hadith
Sacred palces	Jerusalem, Betlehem church, ~~graveyard~~	Mecca, Medina mosque, Ka'ba
Sacred times	Cristmas, Easter Sunday, the Pentacostal Day	Ramadan Friday
Recommended acts	Praying reading Bible	Praying, pilgrimage (haji) alms-giving, reading Qur'an visiting mosque fasting durin Ramadan
Acts that are not recommended	Sin, adultery violence ~~working on Sunday~~ ~~obeying parents~~ ~~not admiring something that belongs to someone else~~ obeying God ~~and not mentioning his name without a reason~~	Drinking alcohol ~~eating pig~~ ~~relationships between opposited sexes (if not relative = adultery)~~

C.2.3 The Data Contributed by Expert III

Table C.6: Here the data is presented as cross-comparison involving three religions. The terms that are in fact valid for the triple comparison are in bold face. For the other terms, a single letter in square brackets marks the religion to the comparison with which they are relevant, whenever confusion might arise. Terms, term parts and similarity aspects that were not used in the evaluation process are marked by strikethrough line.

	Christianity	**Islam**	**Judaism**
Monotheism	**God** **~~You shall have~~ no other gods ~~before Me~~** **Idolatry**	**Allah,** **La ilaha illa Allah** **(No god but God)** **Shirk (=idolatry)**	**God, Yahweh** **~~Elohim~~** **~~You shall have~~ no other gods ~~before Me~~** **Idolatry**
Scripture	**Bible, New Testament Gospels, Revelation**	**Qur'an, Revelation (wahy, tanzil)**	**Torah, Tanakh Bible, Revelation**
Holy community	**Church**	**Ummah**	**~~Holy people (Am Qadosh)~~** **Israel (Yisrael)**
Prophets/ prophecy	**Prophets** Moses (Moshe) [j] Elijah [j], Isaiah [j] Jeremiah [j] ~~"The Twelve" [j]~~	**Prophet (nabi)** **Messenger (rasul)** **Muhammad**	**Prophets** Elijah [c], Isaiah [c] Jeremiah [c] Moses (Moshe) [i]
Sacred Land	**Jerusalem, Rome**	**Mecca, Medina, Jerusalem**	**Israel** **Jerusalem**
Prayer and ritual	**Prayer,** mass [i] eucharist [i] pilgrimage [i] ~~rosary [j]~~	**Prayer, Salat (namaz) du'a,** Hijrah [c] ~~Ziyarah [c]~~ Pilgrimage [c]	**Prayer,** **~~daven~~**
Holy days, times	**Christmas** **~~Epiphany~~** **Good Friday** **Easter** **Pentecost**	**Ramadan** **'Id** **'Id al-Fitr** **~~'Id al-Adha (Qurban Bairam, etc.)~~**	**Sabbath (Shabbat)** **Yom Kippur** **Rosh ha-Shanah**
Denominat-ional, sectarian divisions	**Roman Catholic** **Eastern Orthodox** **Protestant** **~~Arian~~** **~~Monophysite~~** **~~Nestorian~~**	**Sunni** **Shi'i (Shi'ite)** **~~Twelver Shi'I~~** **~~Isma'ili (Sevener Shi'i)~~** **~~Khariji (Kharijite)~~**	**Rabbanite, ~~Karaite~~** **Hasidic, ~~Mitnagid~~** **Reform (Progress, Liberal), Orthodox, Conservative** **Reconstructionist**
Mediator(s), viz between God and humans	Jesus Christ incarnation atonement saints intercession	Muhammad Wali (pl. Awliya' = "saints) Ali, ~~Husayn~~ Imams, intercession	
Atonement	Jesus Christ, Lamb ~~of God~~, ~~Substitutionary~~ atonement, repentance confession ~~absolution~~(sacrament)		Temple ~~sin offering~~ prayer repentence Day of Atonement
Emphasis on law		Shari'ah, fiqh	Torah, halakah
~~Circumcision~~		~~Circumcision~~	Circumcision Bris (Brit)
~~Holy war~~	**~~Crusade~~**, **~~Just War~~**	**Jihad**	**~~Holy War~~**

Appendix D: Proofs for Chapter 5

Lemma 5.1:

(A) At any iterative cycle of the ID algorithm (Figure 5.1) with $t > 0$, update step ID1 decreases the value of F^{ID} (Eq. 5.1) by

$$\Delta F^{ID1}{}_t \;=\; \textstyle\sum_x p(x) KL[p_{t-1}(c|x) \| p_t(c|x)] \,. \tag{D.1}$$

(B) (following Gilad-Bachrach, Navot & Tishby, 2003) At the iterative cycle of the ID algorithm with any t, update step ID2 decreases the value of F^{ID} by

$$\Delta F^{ID2}{}_t \;=\; \beta \textstyle\sum_c p_t(c) KL[p_t(y|c) \| p_{t-1}(y|c)] \,. \tag{D.2}$$

Proof:

(A) Taking log of both sides of the equality sign in step ID1 (note that $p_t(c|x)$ is never equal to 0), we have:

$$\log p_t(c|x) \;=\; \beta \textstyle\sum_y p(y|x) \log p_{t-1}(y|c) - \log z_t(x,\beta) \,, \tag{D.3}$$

where $z_t(x,\beta)$ is a normalization function, over all values c of C, of terms depending on x, c and β. Extracting $\log z_t(x,\beta)$ from the above equality:

$$\log z_t(x,\beta) \;=\; \beta \textstyle\sum_y p(y|x) \log p_{t-1}(y|c) - \log p_t(c|x) \,. \tag{D.4}$$

Note that although a particular value c is being used in Eq. D.4 (which is in fact true for every c), the actual value of $\log z_t(x,\beta)$ does not depend on any particular value of C.

After performing update step ID1 at time t, which results in the replacement of each $p_{t-1}(c|x)$ with $p_t(c|x)$, the value of F^{ID} changes from $F^{ID}{}_{t-1}$, where all $p(c|x)$ and $p(y|c)$ indexed by $t-1$, to

$$F^{ID}{}_{t-} \;\equiv\; \textstyle\sum_x p(x) \sum_c p_t(c|x) \log p_t(c|x) \;-\; \beta \sum_x p(x) \sum_c p_t(c|x) \sum_y p(y|x) \log p_{t-1}(y|c) \,. \tag{D.5}$$

The value we are interested in, $\Delta F^{ID1}{}_t$, is the difference between $F^{ID}{}_{t-1}$ and $F^{ID}{}_{t-}$:

$$\Delta F^{ID1}{}_t \;=\; F^{ID}{}_{t-1} - F^{ID}{}_{t-} \;\;=^{(a)} \tag{D.6}$$

$$\textstyle\sum_x p(x) \sum_c p_{t-1}(c|x) \log p_{t-1}(c|x) \;-\; \beta \sum_x p(x) \sum_c p_{t-1}(c|x) \sum_y p(y|x) \log p_{t-1}(y|c) \;+$$
$$- \textstyle\sum_x p(x) \sum_c p_t(c|x) \left(\beta \sum_y p(y|x) \log p_{t-1}(y|c) - \log z_t(x,\beta) \right)$$
$$+ \;\beta \textstyle\sum_x p(x) \sum_c p_t(c|x) \sum_y p(y|x) \log p_{t-1}(y|c) \;\;=^{(b)}$$

$$\textstyle\sum_x p(x) \sum_c p_{t-1}(c|x) \log p_{t-1}(c|x) \;-\; \beta \sum_x p(x) \sum_c p_{t-1}(c|x) \sum_y p(y|x) \log p_{t-1}(y|c) \;+$$
$$+ \textstyle\sum_x p(x) \sum_c p_{t-1}(c|x) \left(\log z_t(x,\beta) \right) \;\;=^{(c)}$$

$$\sum_x p(x) \sum_c p_{t-1}(c|x) \log p_{t-1}(c|x) \;-\; \beta\sum_x p(x) \sum_c p_{t-1}(c|x) \sum_y p(y|x) \log p_{t-1}(y|c) \;+$$

$$+ \sum_x p(x) \sum_c p_{t-1}(c|x) \left(\beta\sum_y p(y|x) \log p_{t-1}(y|c) - \log p_t(c|x) \right) \overset{(d)}{=}$$

$$\sum_x p(x) KL[p_{t-1}(c|x) \| p_t(c|x)]$$

In the equality chain of Eq. D.6, (a) incorporates Eq. D.3, (b) omits identical terms with opposite signs and replaces, for each x separately, expectation over $p_t(c|x)$ with the identical expectation over $p_{t-1}(c|x)$ (as $\log z_t(x,\beta)$ is independent in C), (c) incorporates Eq. D.4 and (d) again omits opposite sign terms and resorts to the definition of KL divergence.

(B) The value we are interested in, $\Delta F^{ID2}{}_t$, is the difference between $F^{ID}{}_{t-}$ (Eq. D.5) and $F^{ID}{}_t$ (where all $p(c|x)$ and $p(y|c)$ are indexed with t):

$$\Delta F^{ID2}{}_t \;=\; F^{ID}{}_{t-} - F^{ID}{}_t \;\overset{(a)}{=} \tag{D.7}$$

$$-\beta\sum_x p(x) \sum_c p_t(c|x) \sum_y p(y|x) \log p_{t-1}(y|c) + \beta\sum_x p(x) \sum_c p_t(c|x) \sum_y p(y|x) \log p_t(y|c) \;\overset{(b)}{=}$$

$$\beta\sum_{c,y} \left(\log p_t(y|c) - \log p_{t-1}(y|c) \right) \sum_x p(x)\, p_t(c|x)\, p(y|x) \;\overset{(c)}{=}$$

$$\beta\sum_{c,y} \left(\log p_t(y|c) - \log p_{t-1}(y|c) \right) p_t(c,y) \;\overset{(d)}{=}$$

$$\beta\sum_c p_t(c) KL[p_t(y|c) \| p_{t-1}(y|c)]$$

In the equality chain of Eq. D.7, (a) drops the term $\sum_x p(x) \sum_c p_t(c|x) \log p_t(c|x)$ with opposite signs from both $F^{ID}{}_{t-}$ and $F^{ID}{}_t$, (b) just re-orders the terms, (c) uses the conditional independence assumption (Eq. 5.4), which step ID2 happens to maintain, and (d) resorts to the definition of (conditioned) KL divergence.

Lemma 5.2: Stable points of the ID algorithm (i.e. probability distributions that remain unchanged under the update steps: $p_{t+1}(c|x) = p_t(c|x)$ and $p_{t+1}(y|c) = p_t(y|c)$ for all c, x and y) are locally extremal points of F^{ID} (Eq. 5.1).

Proof: Update step ID1 of the ID algorithm can be derived, using the method of *Lagrange multipliers*, as follows:

(1) Convert F^{ID} (Eq. 5.1) to a *Lagrangian* L^{ID1}, by adding to it a Lagrange multiplier λ_x for each x, in order to restrict each probability $p(c|x)$ distribution to sum up to 1: $L^{ID1} = F^{ID} + \sum_x \lambda_x (1 - \sum_c p(c|x))$.

(2) Take derivatives from L^{ID1} with respect to each $p(c|x)$.

(3) Equate each of the resulting terms to 0, extract $p(c|x)$ and set λ_x so that all distributions sum up to 1, to obtain the equation specifying ID1.

The details of this derivation closely resemble the derivation of step IB1 for the IB algorithm (Fig. 5.2), which has been given in several previous works (e.g., Tishby, Pereira & Bialek 1999). The above holds as well with regard to the derivation of update step ID2, substituting $p(c|x)$ by $p(y|c)$, λ_x by λ_c and L^{ID1} by $L^{ID2} = F^{ID} + \sum_c \lambda_c (1 - \sum_y p(y|c))$:

(1) The Lagrangian introducing to F^{ID} the constraint of $p(y|c)$ to sum up to 1 is:

$$L^{ID2} \equiv \sum_x p(x) \sum_c p(c|x) \left(\log p(c|x) - \beta \sum_y p(y|x) \log p(y|c) \right) + \sum_c \lambda_c \left(1 - \sum_y p(y|c) \right). \qquad \text{(D.8)}$$

(2) From L^{ID2}, take derivatives relatively to $p(y|c)$:

$$\frac{\delta L^{ID2}}{\delta p(y\,|\,c)} = -\beta \sum_x p(x)\, p(c|x)\, p(y|x) \left(1 / p(y|c) \right) + \lambda_c . \qquad \text{(D.9)}$$

(3) Equating the above term to 0 and setting $\lambda_c = \beta \sum_x p(x)\ p(c|x) = \beta p(c)$ so that the constraint of $p(y|c)$ to sum up to 1 holds (note that $p(c)$ have here the mere role of a normalization factor), we get the equation underlying step ID2 of the ID algorithm:

$$p(y|c) = \left(1 / p(c) \right) \sum_x p(x)\ p(c|x)\, p(y|x) . \qquad \text{(D.10)}$$

From the above follows that stable probability distributions $p_t(c|x)$ (i.e., ones satisfying $p_{t+1}(c|x) = p_t(c|x)$) specify extremal value of F^{ID} relatively to fixed $p_t(y|c)$ and vice versa: stable probability distributions $p_t(y|c)$ (satisfying $p_{t+1}(y|c) = p_t(y|c)$) specify extremal value of F^{ID} relatively to fixed $p_t(c|x)$. As the $p_t(c|x)$ and $p_t(y|c)$ are all the parameters and they are all fixed in a stable point, together they form an extremal point of F^{ID}.

Lemma 5.7: In the update cycle of time t, the four CP algorithm steps CP1, CP2, CP*1, CP*2, decrease the value of F^{CP1} (Eq. 5.17), F^{CP2} (Eq. 5.19), $F^{\mathrm{CP*1}}$ (Eq. 5.24), $F^{\mathrm{CP*2}}$ (Eq. 5.27) by

$$\begin{aligned}
\Delta F^{CP1}{}_t &= \sum_x p(x) KL[p_{t-1}(c|x) \| p_t(c|x)] , \qquad \text{(D.11)}\\
\Delta F^{CP2}{}_t &= \sum_x p_t(c,w) KL[p_t(y|c,w) \| p_{t-1}(y|c,w)] ,\\
\Delta F^{\mathrm{CP*1}}{}_t &= \sum_y p(y) KL[p^*{}_{t-1}(c|y) \| p^*{}_t(c|y)] ,\\
\Delta F^{\mathrm{CP*2}}{}_t &= \sum_y p^*{}_t(c) KL[p^*{}_t(y|c) \| p^*{}_{t-1}(y|c)] ,
\end{aligned}$$

respectively.

Proof: We exemplify the proof by proving the claim with regard to $F^{\mathrm{CP*1}}$ (note the similarity to the proof of lemma 5.1 (A)).

Taking log of both sides of the step CP*1 equality sign, we have:

$$\log p^*_t(c|y) \ = \ \eta \textstyle\sum_w p(w) \log p_{t-1}(y|c,w) - \log z^*_t(y,\eta)\,, \qquad \text{(D.12)}$$

where $z^*_t(y,\eta)$ is a normalization function over all values c of C of terms depending on y, c and η. We then extract $\log z^*_t(y,\eta)$ from the above equality:

$$\log z^*_t(y,\eta) \ = \ \eta \textstyle\sum_w p(w) \log p_{t-1}(y|c,w) - \ \log p^*_t(c|y)\,. \qquad \text{(D.13)}$$

Note that although a particular value c is being used in Eq. D.13 (in fact, it is true for every c), the actual value of $\log z^*_t(y,\eta)$ does not depend on any particular value of C.

After performing update step CP*1 at time t, which results in the replacement of each $p^*_{t-1}(c|y)$ with $p^*_t(c|y)$, the value of $F^{\text{CP*1}}$ changes from $F^{\text{CP*1}}_{t-1}$, where all $p^*(c|y)$ and $p(y|c,w)$ are indexed by $t-1$, to

(D.14)

$$F^{\text{CP*1}}_{t-} \ \equiv \ \textstyle\sum_y p(y) \sum_c p^*_t(c|y) \log p^*_t(c|y) \ - \ \eta \sum_w p(w) \sum_y p(y) \sum_c p^*_t(c|y) \log p_{t-1}(y|c,w)\,.$$

The value we are interested in, $\Delta F^{\text{CP*1}}_t$, is the difference between $F^{\text{CP*1}}_{t-1}$ and $F^{\text{CP*1}}_{t-}$:

$$\Delta F^{\text{CP*1}}_t \ = \ F^{\text{CP*1}}_{t-1} - F^{\text{CP*1}}_{t-} \ =^{(a)} \qquad \text{(D.15)}$$

$$\textstyle\sum_y p(y) \sum_c p^*_{t-1}(c|y) \log p^*_{t-1}(c|y) \ - \ \eta \sum_w p(w) \sum_y p(y) \sum_c p^*_{t-1}(c|y) \log p_{t-1}(y|c,w) \ +$$

$$- \textstyle\sum_y p(y) \sum_c p^*_t(c|y) \ (\ \eta \sum_y p(y|x) \log \ p_{t-1}(y|c,w) - \log z^*_t(y,\eta)\ )$$

$$+ \ \eta \textstyle\sum_w p(w) \sum_y p(y) \sum_c p^*_t(c|y)\, p_{t-1}(y|c,w) \ =^{(b)}$$

$$\textstyle\sum_y p(y) \sum_c p^*_{t-1}(c|y) \log p_{t-1}(c|x) \ - \ \eta \sum_w p(w) \sum_y p(y) \sum_c \ p^*_{t-1}(c|y) \log \ p_{t-1}(y|c,w) \ +$$

$$+ \textstyle\sum_y p(y) \sum_c p^*_{t-1}(c|y) \ (\ \log z^*_t(y,\eta)\ ) \ =^{(c)}$$

$$\textstyle\sum_y p(y) \sum_c p^*_{t-1}(c|y) \log p^*_{t-1}(c|y) \ - \ \eta \sum_w p(w) \sum_y p(y) \sum_c p^*_{t-1}(c|y) \log \ p_{t-1}(y|c,w) \ +$$

$$+ \textstyle\sum_y p(y) \sum_c p^*_{t-1}(c|y) \ (\ \eta \sum_w p(w) \log p_{t-1}(y|c,w) - \log \ p^*_t(c|y)\ ) \ =^{(d)}$$

$$\textstyle\sum_y p(y)\ KL[\ p^*_{t-1}(c|y) \parallel p^*_t(c|y)\ ]$$

In the equality chain of Eq. D.15, (a) incorporates Eq. D.12, (b) drops identical terms with opposite signs and replaces, for each y separately, expectation over $p^*_t(c|y)$ with the identical expectation over $p^*_{t-1}(c|y)$ (as $\log z^*_t(y,\eta)$ is independent of C), (c) incorporates Eq. D.13 and (d) again drops opposite sign terms and resorts to the definition of KL divergence.

The proof for the claim regarding update step CP1 is in close correspondence to the proof of lemma 5.1 (A). The proofs for the claims regarding update step CP2 and CP*2 are similar to the proof of lemma 5.1 (B).

Lemma 5.8: A set of probability distributions that form a stable point of the CP algorithm (i.e., ones that satisfy $p_{t+1}(c|x) = p_t(c|x)$, $p_{t+1}(y|c,w) = p_t(y|c,w)$, $p^*_{t+1}(c|y) = p^*_t(c|y)$ and $p^*_{t+1}(y|c) = p^*_t(y|c)$, for all c, x, y and w) specifies locally extremal points for F^{CP1} with respect to $p(c|x)$ ($p^*(y|c)$ held fixed), F^{CP2} with respect to $p(y|c,w)$ ($p(c|x)$ held fixed), $F^{\mathrm{CP*1}}$ with respect to $p^*(c|y)$ ($p(y|c,w)$ held fixed) and $F^{\mathrm{CP*2}}$ with respect $p^*(y|c)$ ($p^*(c|y)$ held fixed).

Proof: As a demonstrative example, we show that distributions $p^*(c|y)$ that are part of a stable point of the CP algorithm specify locally extremal points for $F^{\mathrm{CP*1}}$ (while $p(y|c,w)$ held fixed). The other parts are similar (see also the proof of Lemma 5.2 above).

We first write explicitly $L^{\mathrm{CP*1}}$, the Lagrangian introducing to $F^{\mathrm{CP*1}}$ for every y the constraint of $p^*(c|y)$ to sum up to 1:

$$L^{\mathrm{CP*1}} \equiv \tag{D.16}$$

$$\textstyle\sum_y p(y) \sum_c p^*(c|y) \left(\log p^*(c|y) - \eta \sum_w p(w) \log p(y|c,w) \right) + \sum_y \lambda^*_y \left(1 - \sum_c p^*(c|y) \right).$$

From $L^{\mathrm{CP*1}}$, we take derivatives relatively to $p^*(c|y)$, considering $p(y|c,w)$ as a constant:

$$\frac{\delta L^{CP*1}}{\delta p^*(c \mid y)} = p(y) \left(\log p^*(c|y) - \eta \textstyle\sum_w p(w) \log p(y|c,w) \right) + p(y)\ p^*(c|y) \left(1 / p^*(c|y) \right) + \lambda^*_y\,. \tag{D.17}$$

Equating the above term to 0 and setting $\lambda^*_y = p(y) \left(\log z^*(y,\eta) - 1 \right)$, so that the constraint of $p^*(c|y)$ to sum up to 1 holds, with a normalization factor $z^*(y,\eta) = \sum_{c'} \prod_w p(y|c',w)^{\eta\, p(w)}$, we get the equation underlying step IB2 of the IB algorithm:

$$p^*(c|y) = \left(1 / z^*(y,\eta) \right) \textstyle\prod_w p(y|c,w)^{\eta\, p(w)}\,. \tag{D.18}$$

Appendix E: Examples of Religion Cross Partition Clustering

Cross partition clustering configurations produced with the CP plain algorithm (Chapter 5, Figure 5.5; η = 0.4)] with all five religion subsets clustered ($|W| = 5$). Cluster titles were assigned by the authors. The relative size of each cluster according to the *p* probability distribution, as well as to *p**, is indicated. Each keyword *x* appears in the cluster *c* with highest $p(c|x)$, which is indicated in brackets. Along with the elements of every cluster we also indicate the 40 most prominent features (highest $p^*(c|y)$, indicated in square brackets).

E.1: Two Clusters

CLUSTER 1 "The Spiritual Aspect of Religion"

$p(c) = 0.5217$, $p^*(c) = 0.496$

Buddhism: grief(0.998) lamentation(0.997) pain(0.996) sentient(0.994) hell(0.992) birth(0.988) heaven(0.988) suffer(0.986) impermanence(0.986) nirvana(0.981) realm(0.978) cause(0.977) speech(0.977) animal(0.975) rebirth(0.974) existence(0.970) samsara(0.968) livelihood(0.966) being(0.963) soul(0.962) physical(0.961) death(0.960) unwholesome(0.958) water(0.957) anger(0.957) painful(0.954) kill(0.953) god(0.947) consciousness(0.946) karma(0.943) attain(0.942) deliverance(0.942) freedom(0.937) perception(0.931) man(0.928) experience(0.927) noble(0.925) human(0.921) deva(0.921) sense(0.920) enlightenment(0.919) buddhahood(0.916) dukkha(0.907) peace(0.907) phenomenon(0.907) wisdom(0.907) eat(0.905) buddha-nature(0.902) universe(0.901) salvation(0.896) intention(0.890) emotion(0.887) awareness(0.883) awaken(0.883) liberation(0.861) concentration(0.861) noble-truths(0.859) exist(0.849) root(0.846) child(0.844) eightfold-path(0.841) purity(0.828) hear(0.812) mindfulness(0.812) son(0.805) emptiness(0.790) law(0.779) sit(0.753) spirit(0.744) strength(0.718) generosity(0.716) skilful(0.710) tree(0.698) humanity(0.685) element(0.685) bodhisattva(0.682) prince(0.672) sacrifice(0.663) siddhartha-gautama(0.659) conditioning(0.653) friend(0.652) meditator(0.616) teach(0.612) buddha(0.607) word(0.594) learn(0.568) karmic-law(0.567) bodhi-tree(0.567) faith(0.565) problem(0.547) disciple(0.530)

Christianity: flesh(0.996) sin(0.996) heaven(0.996) perish(0.996) sinner(0.995) jesus-christ(0.995) blood(0.994) righteousness(0.994) father(0.994) savior(0.994) forgiveness(0.993) sinful(0.993) reign(0.992) soul(0.991) love(0.991) earth(0.991) holy-spirit(0.991) hell(0.989) punishment(0.989) death(0.988) fire(0.988) love-of-god(0.988) reward(0.987) suffer(0.987) judgment(0.986) redemption(0.985) sacrifice(0.984) god(0.983) kingdom(0.982) water(0.982) bless(0.981) baptize(0.981) devil(0.980) man(0.980) sake(0.978) eye(0.978) human(0.977) sinless(0.977) angel(0.977) grant(0.977) salvation(0.976) humanity(0.975) gift(0.974) eat(0.972) voice(0.972) resurrection(0.972) repentance(0.972) jesus(0.971) mankind(0.970) child(0.967) peace(0.966) pay(0.959) animal(0.957) confess(0.957) lamb(0.957) command(0.956) being(0.954) earthly(0.953) abraham(0.953) godly(0.950) condemnation(0.948) believing(0.947) lost(0.946) mother(0.945) adultery(0.943) divinity(0.941) face(0.939) hear(0.937) good-work(0.932) relationship(0.927) justification(0.926) bread(0.923) faithful(0.923) atonement(0.919) word(0.903) intercession(0.901) commandment(0.901) baptism(0.899) experience(0.884) law(0.883) win(0.882) woman(0.877) passion(0.868) mary(0.864) people(0.863) moses(0.862) faith(0.861) disciple(0.860) prayer(0.846) spiritual(0.844) listen(0.831) house(0.821) fish(0.814) home(0.809) friend(0.807) believer(0.801) israel(0.787) elect(0.786) guide(0.765) miracle(0.732) incarnation(0.728) messiah(0.722)

instruction(0.722) preach(0.717) teach(0.716) gospel(0.715) prophet(0.712)
no-other-god(0.707) learn(0.702) boy(0.701) find(0.692) obey-god(0.682)
violence(0.682) saint(0.671) moral(0.666) cross(0.663) crucifixion(0.650)
birth(0.629) expression(0.614) revelation(0.610) soul-winning(0.581) family(0.569)
minister(0.549) trinity(0.547) paul(0.537) problem(0.535) meet(0.528)
foundation(0.520) question(0.503) isaiah(0.501) gentile(0.500)

Hinduism: heaven(0.990) atman(0.989) liberation(0.988) rebirth(0.986) soul(0.985)
samsara(0.979) attain(0.978) universe(0.972) consciousness(0.969) brahman(0.967)
karma(0.963) divinity(0.961) earth(0.954) demon(0.951) existence(0.941)
moksha(0.941) sun(0.939) birth(0.935) god(0.929) animal(0.926) water(0.926)
sense(0.915) death(0.908) manifestation(0.900) fire(0.858) purity(0.838)
human(0.836) spirit(0.831) man(0.817) indra(0.813) mother(0.798) sacrifice(0.798)
person(0.790) shiva(0.786) shakti(0.781) being(0.780) worshiper(0.759)
experience(0.749) jnana(0.744) invoke(0.743) freedom(0.730) vishnu(0.730)
kill(0.699) son(0.681) brahma(0.666) element(0.639) food(0.635) prayer(0.631)
chakra(0.630) gift(0.609) word(0.572) darshan(0.552) reincarnation(0.535)
humanity(0.533) krishna(0.531) arjuna(0.524) sita(0.518)

Islam: heaven(0.996) hell(0.993) paradise(0.990) creature(0.990) soul(0.989)
earth(0.989) sin(0.985) angel(0.984) deed(0.982) allah(0.981) universe(0.979)
punishment(0.978) water(0.973) animal(0.973) god(0.973) pain(0.972) bless(0.970)
son(0.966) human(0.958) being(0.951) father(0.950) worship(0.948) existence(0.948)
satan(0.947) spirit(0.943) divinity(0.942) death(0.939) remember(0.939)
mother(0.937) believer(0.937) judgment(0.934) man(0.933) physical(0.932)
command(0.922) guide(0.921) purification(0.919) food(0.912) forbid(0.912)
peace(0.909) believing(0.900) tawhid(0.899) wisdom(0.894) messenger(0.893)
mankind(0.888) consciousness(0.887) jesus(0.885) child(0.876) master(0.872)
intercession(0.869) gabriel(0.868) submission(0.860) life(0.858) witness(0.841)
charity(0.840) kill(0.835) daughter(0.825) husband(0.816) moses(0.815) needy(0.810)
lawful(0.806) spiritual(0.792) abraham(0.789) wife(0.787) prophet(0.775)
muhammad-the-prophet-of-allah(0.775) brother(0.774) prayer(0.765) friend(0.759)
share(0.758) word(0.756) adultery(0.755) commandment(0.755) ishmael(0.748)
judge(0.743) poverty(0.741) humanity(0.740) noah(0.738) faith(0.736) justice(0.716)
exist(0.713) companion(0.675) revelation(0.675) saint(0.670) responsibility(0.660)
idolatry(0.650) house(0.646) people(0.645) enemy(0.637) facing(0.636) woman(0.610)
teach(0.608) fast(0.603) testimony(0.595) marriage(0.582) mohamad(0.581) home(0.568)
fight(0.548) find(0.539) sister(0.536) faithful(0.518) meet(0.509) abu-bakr(0.502)

Judaism: heaven(0.983) universe(0.975) reward(0.975) soul(0.974) hashem(0.974)
angel(0.967) sin(0.966) divine-creation(0.959) god(0.958) adam(0.946)
physical(0.935) divinity(0.935) eat(0.932) animal(0.927) human(0.925) water(0.923)
existence(0.913) command(0.910) spirit(0.907) shechinah(0.906) bless(0.904)
man(0.869) golden-calf(0.860) wisdom(0.856) no-other-god(0.851) salvation(0.843)
death(0.841) judgment(0.840) hide(0.822) wife(0.820) peace(0.818) redemption(0.815)
father(0.805) humanity(0.793) hear(0.788) repentance(0.780) food(0.776) son(0.776)
revelation(0.769) mother(0.766) eden(0.764) spiritual(0.757) commandment(0.753)
kill(0.748) forbid(0.747) exist(0.737) teshuvah(0.734) sacrifice(0.714)
candle(0.700) covenant(0.700) child(0.700) kosher(0.697) holy(0.685) sarah(0.683)
expression(0.670) idolatry(0.664) hell(0.663) abraham(0.663) star(0.662)
pharaoh(0.658) freedom(0.651) jacob(0.637) connect(0.619) meal(0.617) violate(0.603)
experience(0.582) mitzvah(0.569) malchut(0.568) noah(0.565) faithful(0.555)
moses(0.553) relationship(0.544) house(0.521) sefirot(0.516)

MOST PROMINENT FEATURES: infinite[0.660] creature[0.647] forgive[0.636]
heaven[0.636] immortal[0.633] hell[0.633] creator[0.630] mercy[0.629] sorrow[0.629]
finite[0.626] flesh[0.624] earth[0.623] pleasure[0.62] eternity[0.621] drink[0.620]
ghost[0.617] eternally[0.617] womb[0.617] beget[0.616] will[0.616] sin[0.615]
bliss[0.615] eternal[0.61] sleep[0.61] heavenly[0.612] ascend[0.612] pain[0.612]
sinful[0.61] soul[0.611] torment[0.611]

CLUSTER 2 “The Establishment Aspect of Religion”

$p(c) = 0.4783$, $p^*(c) = 0.503$

Buddhism: korea(0.999) asia(0.999) south(0.999) east(0.999) sri-lanka(0.999)
china(0.999) japan(0.999) america(0.999) north(0.999) india(0.999) found(0.998)
tibet(0.998) burma(0.998) pitaka(0.998) scholar(0.998) theravada(0.998)
vietnam(0.998) study(0.997) school(0.997) mahayana(0.997) country(0.997) west(0.996)
literature(0.996) thai(0.996) missionary(0.995) zen(0.995) philosophy(0.994)
author(0.993) history(0.993) canonical(0.993) writing(0.993) pali-canon(0.993)

book(0.993) text(0.993) tradition(0.991) christian(0.991) founder(0.991) chapter(0.989) temple(0.989) write(0.989) veda(0.988) sanskrit(0.986) vinaya(0.986) vajrayana(0.985) trade(0.985) social(0.985) ethic(0.985) discussed(0.984) member(0.984) hinduism(0.984) pilgrimage(0.984) tantra(0.984) nun(0.984) english(0.983) celebration(0.983) dalai-lama(0.983) scripture(0.982) hinayana(0.981) writer(0.981) abhidhamma(0.980) monastery(0.980) role(0.978) religious(0.977) student(0.974) stupa(0.973) lama(0.972) religion(0.970) branch(0.969) discourse(0.968) society(0.966) translate(0.966) sutra(0.962) sacred(0.961) koan(0.961) abbot(0.961) psychologist(0.960) sangha(0.959) ceremony(0.955) argue(0.955) project(0.953) ajahn(0.952) ordain(0.948) establish(0.947) vehicle(0.938) vipassana(0.932) theory(0.932) retreat(0.929) king(0.928) monk(0.920) sakya(0.920) discipline(0.907) authority(0.904) argument(0.899) full-moon(0.898) story(0.896) meet(0.893) bhikkhu(0.890) doctrinal(0.885) robe(0.863) teaching(0.853) foundation(0.853) metaphysical(0.845) mantra(0.842) teacher(0.838) instruction(0.829) precept(0.821) begin(0.816) experiment(0.810) master(0.807) five-precepts(0.802) guide(0.793) forest(0.770) practice(0.767) worship(0.765) amida(0.762) family(0.756) meditation(0.736) question(0.736) focus(0.735) house(0.730) mandala(0.723) moral(0.650) attitude(0.648) reincarnation(0.633) gift(0.615) connect(0.609) training(0.596) find(0.567) dharma(0.535) asceticism(0.525) arhat(0.523) people(0.523) relationship(0.522) samadhi(0.518) behavior(0.509) spiritual(0.505)
Christianity: orthodox(0.999) constantinople(0.999) protestant(0.999) vatican(0.999) west(0.999) university(0.999) german(0.999) catholic(0.999) rome(0.998) council(0.998) orthodoxy(0.998) america(0.998) cardinal(0.998) bishop(0.997) pope(0.997) patriarch(0.996) organization(0.995) found(0.991) baptist(0.990) apostolic(0.989) monk(0.988) evangelical(0.988) school(0.984) theology(0.982) sunday(0.980) greek(0.979) postmodern(0.979) easter(0.977) luther(0.976) tradition(0.976) writing(0.975) church(0.973) writer(0.970) association(0.970) founder(0.968) student(0.967) history(0.965) ancient(0.961) religious(0.960) new-testament(0.960) good-friday(0.957) book(0.954) theory(0.950) trade(0.949) text(0.946) mass(0.943) eucharist(0.941) jew(0.941) role(0.935) chapter(0.931) pilgrimage(0.913) member(0.909) study(0.908) christmas(0.907) thomas(0.903) story(0.895) hebrew(0.892) author(0.887) translate(0.887) ethic(0.876) tabernacle(0.866) pentecost(0.865) sacrament(0.859) city(0.849) old-testament(0.842) religion(0.822) doctrinal(0.817) convert(0.804) luke(0.797) teaching(0.785) matthew(0.769) king(0.764) sabbath(0.754) authority(0.749) apostle(0.744) language(0.740) write(0.736) bethlehem(0.731) argument(0.731) priestly(0.731) bible(0.722) teacher(0.708) jerusalem(0.705) john(0.694) passage(0.688) ministry(0.686) predict(0.680) establish(0.678) idolatry(0.673) service(0.666) john-the-baptist(0.661) worship(0.619) bearing(0.617) jordan-river(0.610) jeremiah(0.595) argue(0.594) refer(0.574) reading-bible(0.568) scripture(0.553) season(0.547) zion(0.531) elijah(0.522)
Hinduism: south(0.999) found(0.999) indus(0.999) scholar(0.998) shaiva(0.998) classical(0.998) aryan(0.997) jain(0.997) literature(0.997) founder(0.997) valley(0.997) muslim(0.996) brahma-sutra(0.996) sanskrit(0.996) author(0.996) india(0.996) christian(0.996) civilization(0.995) history(0.995) west(0.995) festival(0.995) school(0.995) buddhism(0.995) study(0.994) text(0.993) sutra(0.992) ancient(0.992) student(0.992) language(0.992) write(0.992) writing(0.992) book(0.991) ramayana(0.990) poet(0.990) shri(0.989) varanasi(0.989) foundation(0.988) country(0.988) art(0.988) purana(0.988) vaishnavism(0.987) religious(0.987) samkhya(0.987) ahram(0.987) philosophy(0.986) kumbhamela(0.986) science(0.985) epic(0.985) tradition(0.984) social(0.983) mahabharata(0.983) poem(0.981) smriti(0.980) mahatma(0.979) ram(0.979) temple(0.979) story(0.978) agama(0.976) dance(0.975) vedas(0.972) society(0.971) rigveda(0.969) priest(0.968) teacher(0.964) theory(0.963) teaching(0.962) caste(0.959) pilgrimage(0.952) raja(0.951) sankara(0.948) religion(0.948) brahmin(0.946) king(0.944) ceremony(0.938) hymn(0.935) authority(0.934) scripture(0.931) advaitha(0.928) gita(0.920) sadhu(0.916) ritual(0.916) upanishad(0.912) yoga(0.909) doctrine(0.903) practice(0.899) rite(0.891) faith(0.879) buddha(0.873) family(0.871) idea(0.845) puja(0.839) asceticism(0.836) find(0.835) teach(0.833) sacred(0.831) yogi(0.820) learn(0.806) ramanuja(0.804) people(0.801) question(0.797) brahmana(0.794) holy(0.790) holy-people(0.775) hero(0.736) spirituality(0.736) durga(0.719) rama(0.715) star(0.688) trimurti(0.664) guru(0.659) bhakti(0.635) ganesh(0.635) child(0.630) discipline(0.622) dharma(0.611) law(0.596) meditation(0.567) idol(0.563) devotee(0.561) kali(0.527)

Islam: africa(0.999) asia(0.999) east(0.999) university(0.998) india(0.996)
shiah(0.996) empire(0.994) found(0.994) shii(0.993) sunni(0.993) hijrah(0.993)
west(0.993) student(0.990) school(0.989) baghdad(0.988) philosophy(0.988)
arabia(0.987) country(0.987) racial(0.986) founder(0.984) writer(0.984) fiqh(0.983)
economy(0.982) study(0.979) tipu(0.978) id-al-fitr(0.978) jew(0.975)
christian(0.974) author(0.974) bank(0.972) madinah(0.972) city(0.971) arab(0.970)
chapter(0.969) calif(0.969) writing(0.968) civilization(0.968) translate(0.966)
scholar(0.965) trade(0.965) religious(0.965) science(0.964) branch(0.964)
army(0.958) english(0.957) history(0.954) sheikh(0.954) sufi(0.953) istanbul(0.949)
mecah(0.947) polygamy(0.945) finance(0.945) friday(0.944) mosque(0.943)
tradition(0.940) jerusalem(0.940) social(0.933) ritual(0.917) write(0.911)
surah(0.910) imam(0.909) company(0.908) haj(0.908) shariah(0.908) practice(0.903)
foundation(0.897) language(0.895) doctrine(0.893) succession(0.890)
pillars-of-faith(0.888) bible(0.878) kaabah(0.865) story(0.861) ali(0.858)
teacher(0.852) tribe(0.851) teaching(0.839) id(0.825) establish(0.818)
pilgrimage(0.817) sacred(0.816) race(0.810) society(0.804) israel(0.802)
hadith(0.801) scripture(0.798) book(0.793) ummah(0.792) ramadan(0.789) sunnah(0.771)
problem(0.770) religion(0.764) jihad(0.756) umar(0.754) ruler(0.754) bukhari
muslim(0.739) learn(0.730) female(0.718) service(0.713) code(0.705) preach(0.701)
law(0.686) authority(0.667) attitude(0.637) quran(0.622) moral(0.602) salah(0.602)
dua(0.595) question(0.579) mission(0.574) read-quran(0.571) statement(0.567)
freedom(0.566) drink-alcohol(0.564) family(0.542) descendant(0.542) haram(0.522)
declaration(0.521) holy(0.509)

Judaism: europe(0.999) conservative(0.999) reconstructionism(0.999) east(0.999)
sephardim(0.999) ashkenazim(0.999) zionism(0.999) america(0.999) north(0.999)
reform(0.998) orthodox(0.998) german(0.997) yeshivah(0.997) found(0.997)
rabbinical(0.996) philosophy(0.996) school(0.996) founder(0.995) literary(0.995)
calendar(0.994) tanakh(0.993) babylon(0.993) scholar(0.992) mishnah(0.991)
writer(0.990) center(0.990) christian(0.989) liturgy(0.989) synagogue(0.989)
theology(0.988) rome(0.987) country(0.985) ancient(0.984) writing(0.984)
author(0.984) community(0.983) religious(0.983) daniel(0.981) hasid(0.981)
tradition(0.980) kabalah(0.979) student(0.979) city(0.978)
principles-of-faith(0.977) judah(0.977) oral(0.977) bible(0.976) book(0.976)
history(0.976) scripture(0.974) hebrew(0.974) text(0.974) area(0.973) talmud(0.973)
language(0.972) establish(0.971) jerusalem(0.970) religion(0.969) teacher(0.969)
scroll(0.966) siddur(0.965) member(0.964) temple-mount(0.962) festival(0.962)
gemara(0.962) exodus(0.960) rosh-hodesh(0.958) social(0.958) role(0.955) code(0.954)
shavuot(0.951) ari(0.949) chapter(0.948) celebration(0.948) holiday(0.948)
halachah(0.947) society(0.946) authority(0.946) discuss(0.942) rabbi(0.942)
zohar(0.939) teaching(0.939) law(0.938) study(0.935) service(0.933)
rosh-hashanah(0.930) read-torah(0.927) mystic(0.926) kingdom(0.923)
foundation(0.921) tribe(0.909) story(0.901) write(0.898) sukoth(0.893) temple(0.893)
yarmulke(0.893) israel(0.890) intellectual(0.883) argument(0.880) argue(0.880)
levi(0.877) solomon(0.877) letter(0.877) parsha(0.875) purim(0.865) passover(0.859)
bris(0.852) ritual(0.850) mourn(0.848) statement(0.841) sea(0.837) question(0.834)
mikveh(0.827) canaan(0.823) exile(0.811) priest(0.809) yhwh(0.805)
high-priest(0.803) gentile(0.800) tefilin(0.797) jeremiah(0.792) talit(0.776)
ishaiah(0.775) david(0.773) find(0.769) family(0.761) ezekiel(0.760) esther(0.755)
learn(0.753) jesus(0.721) circumcision(0.716) status(0.713) worship(0.711)
sabbath(0.711) wear(0.698) women(0.694) home(0.680) torah(0.676) elijah(0.674)
marriage(0.672) seal(0.668) israelite(0.667) saint(0.658) prayer(0.652)
messiah(0.644) menorah(0.638) atonement(0.629) joseph(0.614) king(0.612)
moral(0.607) faith(0.601) teach(0.585) people(0.534) prophet(0.522) egypt(0.521)
word(0.516) isaac(0.513) mount-sinai(0.504)

MOST PROMINENT FEATURES: eastern[0.711] north[0.701] south[0.695] central[0.688]
asia[0.687] africa[0.684] east[0.683] europe[0.682] southern[0.681]
university[0.678] orthodox[0.676] canada[0.676] coast[0.675] eastern[0.674]
academy[0.673] graduate[0.673] college[0.673] medieval[0.673] professor[0.672]
italy[0.671] reform[0.670] 16th[0.669] european[0.667] official[0.667]
affiliate[0.666] historian[0.666] british[0.664] headquarters[0.662]
edition[0.662] england[0.661]

E.2: Seven Clusters

CLUSTER 1 “Schools” $p(c) = 0.1993$, $p^*(c) = 0.1457$
Buddhism: america(0.999) asia(0.999) japan(0.999) west(0.999) east(0.999) korea(0.999) india(0.999) china(0.999) tibet(0.999) christian(0.999) school(0.999) theravada(0.999) hinduism(0.999) south(0.999) found(0.999) tradition(0.999) study(0.998) country(0.998) history(0.998) philosophy(0.997) religion(0.997) role(0.997) mahayana(0.997) religious(0.996) zen(0.996) sri-lanka(0.995) scholar(0.995) social(0.994) society(0.993) ethic(0.991) burma(0.991) missionary(0.988) founder(0.987) hinayana(0.987) vietnam(0.984) establish(0.982) member(0.981) veda(0.976) argue(0.968) vehicle(0.962) north(0.952) psychologist(0.945) vajrayana(0.944) literature(0.895) tantra(0.895) writing(0.890) authority(0.874) writer(0.859) sangha(0.855) student(0.854) thai(0.843) discussed(0.810) project(0.775) theory(0.745) vinaya(0.696) vipassana(0.676) monastery(0.649) discipline(0.608) author(0.599) begin(0.588) trade(0.579) doctrinal(0.561) koan(0.482)
Christianity: orthodox(0.999) protestant(0.999) catholic(0.999) west(0.999) orthodoxy(0.999) organization(0.999) rome(0.999) council(0.999) america(0.999) pope(0.999) university(0.998) religious(0.998) evangelical(0.997) luther(0.995) theology(0.995) vatican(0.993) german(0.993) constantinople(0.992) bishop(0.992) role(0.991) postmodern(0.991) found(0.988) apostolic(0.985) church(0.981) association(0.977) baptist(0.954) religion(0.936) tradition(0.933) patriarch(0.919) jew(0.890) member(0.861) founder(0.815) authority(0.796) establish(0.746) history(0.707) ethic(0.688) ancient(0.595) cardinal(0.558)
Hinduism: west(0.999) christian(0.999) religious(0.999) civilization(0.999) buddhism(0.999) aryan(0.999) social(0.998) founder(0.998) shaiva(0.998) caste(0.998) society(0.997) south(0.997) found(0.997) indus(0.996) samkhya(0.996) religion(0.996) science(0.996) history(0.996) tradition(0.994) scholar(0.992) foundation(0.991) country(0.989) muslim(0.989) india(0.989) art(0.988) study(0.988) ancient(0.985) theory(0.984) classical(0.979) faith(0.973) philosophy(0.972) authority(0.972) school(0.969) valley(0.958) practice(0.940) teaching(0.921) advaitha(0.912) student(0.899) jain(0.869) doctrine(0.840) idea(0.833) literature(0.832) yoga(0.813) sankara(0.805) language(0.736) poet(0.674)
Islam: africa(0.999) asia(0.999) west(0.999) east(0.999) sunni(0.999) shiah(0.998) christian(0.998) country(0.998) civilization(0.998) philosophy(0.998) racial(0.998) found(0.997) shii(0.997) religious(0.997) economy(0.992) sufi(0.988) university(0.984) science(0.983) jew(0.982) school(0.981) practice(0.979) founder(0.978) history(0.975) empire(0.966) foundation(0.958) study(0.952) race(0.951) india(0.951) student(0.932) finance(0.930) doctrine(0.929) polygamy(0.905) shariah(0.887) religion(0.872) fiqh(0.841) branch(0.822) establish(0.801) teaching(0.794) ummah(0.762) arabia(0.736) scholar(0.685) tradition(0.680) ruler(0.645) jihad(0.638) sheikh(0.633) social(0.566) authority(0.535) society(0.534)
Judaism: reform(0.999) conservative(0.999) reconstructionism(0.999) zionism(0.999) orthodox(0.999) america(0.999) europe(0.999) sephardim(0.999) ashkenazim(0.999) religious(0.999) christian(0.999) east(0.999) religion(0.998) german(0.998) theology(0.997) philosophy(0.997) rabbinical(0.996) community(0.996) role(0.995) authority(0.995) society(0.993) social(0.981) establish(0.978) found(0.976) founder(0.972) country(0.966) history(0.963) tradition(0.953) ancient(0.927) school(0.921) scholar(0.881) halachah(0.875) center(0.870) writer(0.866) principles-of-faith(0.857) argue(0.851) literary(0.836) area(0.820) foundation(0.775) law(0.739) code(0.734) student(0.705) hasid(0.704) kabalah(0.700) north(0.656) liturgy(0.640) teacher(0.600) jesus(0.592) member(0.581) argument(0.571) ritual(0.568) mystic(0.460) rabbi(0.427) messiah(0.281)
MOST PROMINENT FEATURES: eastern[0.350] africa[0.329] europe[0.313] north[0.308] east[0.307] central[0.297] asia[0.294] dominant[0.293] south[0.292] orthodox[0.292] mainstream[0.291] protestant[0.285] affiliate[0.285] coast[0.284] reform[0.284] medieval[0.281] ethnic[0.280] russia[0.280] african[0.28] conservative[0.278] empire[0.278] southern[0.278] 19th[0.277] oriental[0.274] canada[0.274] germany[0.273] west[0.273] political[0.272] minority[0.272]

CLUSTER 2 “Divinity” $p(c) = 0.1876$, $p^*(c) = 0.1379$

Buddhism: god(0.865) brahma(0.854)

Christianity: holy-spirit(0.999) jesus-christ(0.998) god(0.997) father(0.997) savior(0.996) jesus(0.988) baptize(0.988) salvation(0.986) reign(0.985) gift(0.985) believing(0.980) love-of-god(0.973) baptism(0.972) disciple(0.971) command(0.964) confess(0.951) commandment(0.950) messiah(0.932) faith(0.919) resurrection(0.902) abraham(0.837) love(0.835) believer(0.796) teach(0.794) bless(0.787) redemption(0.742) justification(0.741) grant(0.715) forgiveness(0.690) gospel(0.681) law(0.661) moses(0.608) faithful(0.574) word(0.517) hear(0.516) miracle(0.461) being(0.456) prayer(0.438)

Hinduism: god(0.995) brahma(0.497)

Islam: god(0.999) allah(0.998) peace(0.997) messenger(0.991) jesus(0.989) worship(0.986) believing(0.981) tawhid(0.978) command(0.967) abraham(0.964) guide(0.960) prophet(0.954) moses(0.934) bless(0.928) believer(0.902) mohamad(0.863) angel(0.829) mankind(0.784) companion(0.772) divinity(0.741) deed(0.629) teach(0.584)

Judaism: god(0.994) hashem(0.987) bless(0.656) commandment(0.632) abraham(0.336)

MOST PROMINENT FEATURES: omnipotent[0.272] omniscient[0.264] almighty[0.256] mercy[0.240] infinite[0.240] worship[0.238] all-know[0.236] glory[0.234] creator[0.234] grace[0.233] saviour[0.231] manifestation[0.230] ye[0.229] transcendent[0.228] praise[0.226] wrath[0.226] gracious[0.225] bestow[0.222] trinity[0.222] disobey[0.222] sovereign[0.221] jealous[0.219] universe[0.219] remembrance[0.219] pray[0.218] surrender[0.217] forgive[0.217] curse[0.216] attribute[0.215]

CLUSTER 3 “Religious Experience” $p(c) = 0.1638$, $p^*(c) = 0.1434$

Buddhism: phenomenon(0.999) perception(0.999) consciousness(0.999) human(0.999) concentration(0.999) mindfulness(0.999) physical(0.999) livelihood(0.999) liberation(0.999) buddha-nature(0.999) awareness(0.999) freedom(0.999) law(0.999) wisdom(0.999) eightfold-path(0.999) sentient(0.999) emptiness(0.999) purity(0.999) sense(0.999) attain(0.999) experience(0.999) existence(0.999) karma(0.999) buddhahood(0.999) speech(0.998) universe(0.998) soul(0.998) impermanence(0.998) salvation(0.997) emotion(0.997) spiritual(0.997) element(0.997) peace(0.997) noble-truths(0.997) intention(0.997) enlightenment(0.997) nirvana(0.997) moral(0.996) skilful(0.996) behavior(0.996) relationship(0.996) humanity(0.995) conditioning(0.995) exist(0.994) meditator(0.992) awaken(0.991) noble(0.991) generosity(0.990) bodhisattva(0.988) suffer(0.988) cause(0.988) unwholesome(0.987) arhat(0.986) being(0.985) dukkha(0.984) strength(0.983) samadhi(0.981) karmic-law(0.976) connect(0.975) attitude(0.970) root(0.965) deliverance(0.965) faith(0.961) training(0.957) focus(0.954) samsara(0.947) spirit(0.930) rebirth(0.928) problem(0.906) teach(0.897) realm(0.872) practice(0.849) meditation(0.760) dharma(0.754) guide(0.743) painful(0.739) metaphysical(0.731) foundation(0.702) five-precepts(0.693) gift(0.632) anger(0.587) reincarnation(0.577) find(0.540) sacrifice(0.474)

Christianity: moral(0.999) human(0.996) humanity(0.973) spiritual(0.968) relationship(0.961) experience(0.956) expression(0.946) incarnation(0.898) divinity(0.845) atonement(0.799) argue(0.522) guide(0.455)

Hinduism: consciousness(0.999) atman(0.999) human(0.999) existence(0.998) liberation(0.998) jnana(0.998) purity(0.998) sense(0.998) moksha(0.998) soul(0.997) freedom(0.997) attain(0.997) universe(0.996) karma(0.996) experience(0.992) brahman(0.984) humanity(0.983) manifestation(0.978) discipline(0.972) spirit(0.955) element(0.949) rebirth(0.949) reincarnation(0.934) being(0.914) bhakti(0.904) dharma(0.858) law(0.853) spirituality(0.835) divinity(0.790) samsara(0.624)

Islam: spiritual(0.999) human(0.999) physical(0.998) moral(0.997) consciousness(0.994) humanity(0.992) exist(0.992) justice(0.991) life(0.989) existence(0.988) universe(0.973) code(0.938) freedom(0.938) being(0.933) submission(0.922) wisdom(0.916) spirit(0.904) law(0.866) attitude(0.817) purification(0.802) judge(0.797) responsibility(0.705) faith(0.595) commandment(0.577) man(0.513) creature(0.493) problem(0.448)

Judaism: spiritual(0.999) human(0.999) existence(0.999) physical(0.998) expression(0.998) humanity(0.996) experience(0.993) moral(0.992) connect(0.991) revelation(0.989) relationship(0.986) soul(0.983) exist(0.978) freedom(0.971) universe(0.937) malchut(0.923) divinity(0.903) wisdom(0.895) spirit(0.887) intellectual(0.819) no-other-god(0.784) shechinah(0.761) faith(0.758) divine-creation(0.606) sefirot(0.442)
MOST PROMINENT FEATURES: intrinsic[0.281] mental[0.26] realm[0.260] mature[0.259] rights[0.258] subjective[0.258] potential[0.254] intellect[0.253] dimension[0.253] intuitive[0.252] goal[0.251] objective[0.251] innate[0.251] perception[0.246] physical[0.244] species[0.243] material[0.242] finite[0.242] limitation[0.242] emotional[0.241] transcend[0.239] dignity[0.23] perfection[0.238] nature[0.238] infinite[0.237] betterment[0.235] physical[0.233] energy[0.233] reality[0.232] temporal[0.232]

CLUSTER 4 “Writings” $p(c) = 0.1181$, $p^*(c) = 0.1422$
Buddhism: pali-canon(0.999) sanskrit(0.999) sutra(0.999) pitaka(0.999) english(0.998) translate(0.997) chapter(0.994) abhidhamma(0.992) book(0.990) canonical(0.988) write(0.980) discourse(0.977) text(0.962) scripture(0.864) word(0.847) argument(0.643) story(0.639) teaching(0.500) precept(0.346)
Christianity: chapter(0.999) hebrew(0.999) translate(0.999) greek(0.999) new-testament(0.999) book(0.999) text(0.999) old-testament(0.999) luke(0.999) matthew(0.999) passage(0.999) author(0.998) write(0.998) bible(0.998) writer(0.998) john(0.997) writing(0.997) study(0.996) apostle(0.993) isaiah(0.992) scripture(0.991) language(0.983) revelation(0.983) refer(0.981) paul(0.967) teaching(0.951) thomas(0.898) argument(0.896) theory(0.883) instruction(0.767) prophet(0.751) doctrinal(0.662) trinity(0.591) foundation(0.440) birth(0.359)
Hinduism: rigveda(0.999) gita(0.999) sanskrit(0.999) upanishad(0.999) sutra(0.998) smriti(0.998) brahma-sutra(0.996) scripture(0.993) mahabharata(0.990) poem(0.989) text(0.988) purana(0.987) agama(0.987) hymn(0.984) vedas(0.979) epic(0.979) word(0.977) writing(0.977) book(0.974) write(0.966) author(0.814) ramayana(0.639) ramanuja(0.614)
Islam: chapter(0.999) surah(0.999) bible(0.999) write(0.999) translate(0.999) hadith(0.999) book(0.999) language(0.997) scripture(0.996) quran(0.993) statement(0.989) sunnah(0.987) arab(0.985) author(0.972) english(0.965) imam(0.936) word(0.932) bukhari-muslim(0.922) writing(0.903) noah(0.869) holy(0.841) gabriel(0.692) revelation(0.653) writer(0.454) testimony(0.396)
Judaism: tanakh(0.999) scripture(0.999) mishnah(0.999) book(0.999) oral(0.999) talmud(0.999) bible(0.999) write(0.999) letter(0.999) writing(0.999) gemara(0.999) chapter(0.999) word(0.998) zohar(0.997) text(0.996) torah(0.995) hebrew(0.989) author(0.951) ishaiah(0.948) prophet(0.947) language(0.921) siddur(0.918) statement(0.912) exodus(0.866) moses(0.862) scroll(0.852) teaching(0.765) discuss(0.649) study(0.432)
MOST PROMINENT FEATURES: commentary[0.332] manuscript[0.330] translation[0.328] dictionary[0.316] grammar[0.304] translate[0.299] english[0.293] canon[0.289] translator[0.285] compile[0.284] authoritative[0.284] compilation[0.281] written[0.280] script[0.280] collection[0.279] pronunciation[0.277] edition[0.277] edit[0.276] print[0.275] publish[0.274] read[0.273] extant[0.271] verb[0.271] latin[0.270] text[0.270] treatise[0.268] verse[0.267] hebraic[0.266] publisher[0.266]

CLUSTER 5 “Festivals and Rite” $p(c) = 0.1163$, $p^*(c) = 0.1443$

Buddhism: full-moon(0.999) celebration(0.999) stupa(0.999) ceremony(0.999) sakya(0.998) abbot(0.998) ajahn(0.997) robe(0.997) retreat(0.996) house(0.994) forest(0.991) mandala(0.991) temple(0.985) king(0.982) bodhi-tree(0.981) pilgrimage(0.978) worship(0.976) amida(0.971) ordain(0.949) mantra(0.939) bhikkhu(0.922) monk(0.897) nun(0.887) experiment(0.882) sit(0.870) asceticism(0.806) meet(0.792) dalai-lama(0.744) branch(0.726) sacred(0.551) instruction(0.548) prince(0.531) lama(0.528) siddhartha-gautama(0.459)

Christianity: easter(0.999) tabernacle(0.999) christmas(0.999) sunday(0.999) sabbath(0.999) jerusalem(0.999) pentecost(0.999) city(0.998) season(0.998) eucharist(0.996) pilgrimage(0.995) zion(0.994) bethlehem(0.993) story(0.992) jordan-river(0.991) service(0.991) trade(0.974) elijah(0.970) good-friday(0.969) mass(0.959) sacrament(0.955) idolatry(0.930) worship(0.913) john-the-baptist(0.910) jeremiah(0.908) ministry(0.904) king(0.892) house(0.854) bearing(0.826) priestly(0.817) minister(0.812) monk(0.810) meet(0.767) saint(0.752) school(0.742) israel(0.716) convert(0.633) predict(0.611) no-other-god(0.600) intercession(0.545) soul-winning(0.512) student(0.439) gentile(0.403)

Hinduism: puja(0.999) ganesh(0.999) festival(0.999) ceremony(0.999) durga(0.999) rama(0.999) pilgrimage(0.999) rite(0.999) temple(0.999) holy(0.999) king(0.999) kali(0.998) priest(0.998) sacred(0.997) darshan(0.996) kumbhamela(0.995) prayer(0.995) devotee(0.994) idol(0.993) ahram(0.993) sita(0.993) invoke(0.991) ram(0.988) sadhu(0.988) varanasi(0.984) krishna(0.984) holy-people(0.981) hero(0.971) dance(0.971) ritual(0.968) brahmana(0.967) shri(0.944) gift(0.939) shiva(0.930) raja(0.927) worshiper(0.926) arjuna(0.920) star(0.913) brahmin(0.896) trimurti(0.874) story(0.841) asceticism(0.840) vishnu(0.837) yogi(0.829) buddha(0.818) vaishnavism(0.802) guru(0.753) meditation(0.678) shakti(0.562) chakra(0.558) mahatma(0.546)

Islam: kaabah(0.999) id(0.999) ramadan(0.999) friday(0.999) id-al-fitr(0.999) haj(0.999) mecah(0.999) mosque(0.999) salah(0.999) pilgrimage(0.999) jerusalem(0.999) hijrah(0.999) madinah(0.998) city(0.996) istanbul(0.995) ishmael(0.995) pillars-of-faith(0.994) baghdad(0.994) ritual(0.992) fast(0.986) umar(0.986) army(0.985) story(0.984) house(0.984) succession(0.979) dua(0.976) company(0.968) prayer(0.961) israel(0.941) faithful(0.929) sacred(0.918) tipu(0.917) ali(0.887) service(0.859) trade(0.856) calif(0.843) tribe(0.835) descendant(0.814) abu-bakr(0.761) bank(0.756) charity(0.739) saint(0.726) read-quran(0.699) meet(0.520) muhammad-the-prophet-of-allah(0.514) mission(0.447) preach(0.280) declaration(0.232)

Judaism: sukoth(0.999) festival(0.999) shavuot(0.999) temple(0.999) passover(0.999) jerusalem(0.999) rosh-hashanah(0.999) temple-mount(0.999) rosh-hodesh(0.999) celebration(0.999) high-priest(0.999) sabbath(0.999) atonement(0.999) holiday(0.999) mourn(0.999) purim(0.999) menorah(0.999) city(0.999) canaan(0.999) mikveh(0.999) candle(0.999) priest(0.998) read-torah(0.998) service(0.998) david(0.997) calendar(0.997) solomon(0.995) tribe(0.993) king(0.993) house(0.993) kingdom(0.992) exile(0.992) levi(0.991) talit(0.991) meal(0.990) sea(0.989) elijah(0.986) bris(0.985) rome(0.980) holy(0.979) esther(0.976) judah(0.972) worship(0.972) yhwh(0.971) yarmulke(0.970) prayer(0.969) saint(0.967) seal(0.960) israelite(0.958) synagogue(0.958) babylon(0.956) ari(0.952) parsha(0.945) egypt(0.944) isaac(0.939) wear(0.934) sacrifice(0.932) tefilin(0.916) circumcision(0.912) israel(0.902) ezekiel(0.871) covenant(0.791) noah(0.783) faithful(0.706) mount-sinai(0.701) yeshivah(0.701) daniel(0.698) home(0.652) jeremiah(0.641) star(0.595) story(0.586) repentance(0.580) pharaoh(0.496) mitzvah(0.363)

MOST PROMINENT FEATURES: annual[0.328] festival[0.308] saturday[0.298] friday[0.294] funeral[0.286] rebuild[0.285] feast[0.282] grand[0.281] decorate[0.281] shrine[0.281] holiday[0.278] bus[0.278] noon[0.276] commemorate[0.273] celebrated[0.270] yearly[0.270] mile[0.269] congregational[0.269] dome[0.268] pilgrimage[0.267] memorial[0.267] nearby[0.26] season[0.264] don[0.262] afternoon[0.262] sunday[0.261] december[0.260] evening[0.260] monday[0.259] throng[0.259]

CLUSTER 6 “Sin, Suffering and Material Existence” $p(c) = 0.1095$, $p^*(c) = 0.1425$
Buddhism: lamentation (0.999) water (0.999) grief (0.999) kill (0.999) eat (0.999) hell (0.997) animal (0.992) death (0.982) heaven (0.982) birth (0.962) pain (0.919) man (0.862) tree (0.755) deva (0.525)
Christianity: fire (0.999) punishment (0.999) eat (0.999) water (0.999) animal (0.999) lost (0.998) hell (0.998) perish (0.997) lamb (0.997) pay (0.996) blood (0.995) eye (0.993) condemnation (0.993) fish (0.991) sinless (0.989) adultery (0.989) violence (0.989) suffer (0.984) soul (0.977) godly (0.976) earth (0.971) face (0.971) reward (0.966) death (0.961) devil (0.958) flesh (0.957) sin (0.949) sinful (0.946) judgment (0.943) win (0.934) sake (0.933) passion (0.915) mankind (0.899) sinner (0.896) man (0.891) heaven (0.872) voice (0.863) bread (0.857) cross (0.844) earthly (0.833) good-work (0.798) sacrifice (0.757) crucifixion (0.695) obey-god (0.673) righteousness (0.584) angel (0.563) peace (0.525) repentance (0.518) kingdom (0.510) elect (0.506)
Hinduism: animal (0.999) heaven (0.999) earth (0.998) death (0.998) water (0.997) kill (0.996) demon (0.992) birth (0.958) sun (0.955) food (0.941) man (0.825) fire (0.634) sacrifice (0.569) indra (0.537) person (0.413)
Islam: water (0.999) animal (0.999) hell (0.999) punishment (0.999) paradise (0.998) food (0.998) pain (0.997) sin (0.995) earth (0.995) adultery (0.994) kill (0.993) death (0.993) lawful (0.990) satan (0.986) heaven (0.984) forbid (0.981) poverty (0.946) soul (0.910) enemy (0.836) intercession (0.832) facing (0.818) drink-alcohol (0.754) haram (0.723) judgment (0.622) needy (0.606) fight (0.527) idolatry (0.509) remember (0.499) witness (0.449) master (0.443)
Judaism: animal (0.999) water (0.998) eat (0.998) kosher (0.998) sin (0.989) heaven (0.989) death (0.987) food (0.987) forbid (0.986) idolatry (0.966) eden (0.955) hell (0.952) kill (0.951) angel (0.946) adam (0.913) violate (0.887) reward (0.871) hide (0.820) man (0.746) command (0.708) redemption (0.624) peace (0.591) teshuvah (0.590) judgment (0.579) golden-calf (0.534) salvation (0.467)
MOST PROMINENT FEATURES: vegetable [0.321] insect [0.318] penalty [0.308] quench [0.303] drink [0.301] plant [0.296] meat [0.294] fish [0.294] hell [0.293] bird [0.284] slaughter [0.281] beast [0.278] drop [0.278] taste [0.276] worm [0.274] drinking [0.274] wild [0.273] pig [0.273] grass [0.272] torment [0.268] boil [0.267] pour [0.266] smoke [0.266] poison [0.263] glass [0.263] specie [0.261] fat [0.261] decay [0.26] injure [0.259]

CLUSTER 7 “Community and Family” $p(c) = 0.1053$, $p^*(c) = 0.144$
Buddhism: child (0.996) friend (0.993) son (0.981) people (0.977) family (0.958) question (0.940) learn (0.881) hear (0.810) teacher (0.808) disciple (0.528) master (0.470)
Christianity: friend (0.992) family (0.989) mother (0.984) boy (0.982) question (0.957) woman (0.956) problem (0.941) learn (0.929) child (0.908) home (0.906) listen (0.893) teacher (0.834) preach (0.794) find (0.759) people (0.723) reading-bible (0.423) mary (0.342)
Hinduism: child (0.999) question (0.997) son (0.994) mother (0.976) family (0.974) learn (0.958) people (0.911) teacher (0.713) teach (0.615) find (0.606)
Islam: sister (0.999) husband (0.999) wife (0.999) child (0.997) family (0.997) marriage (0.996) mother (0.996) woman (0.993) brother (0.987) question (0.987) father (0.985) daughter (0.985) friend (0.979) female (0.934) home (0.921) share (0.875) find (0.859) people (0.849) teacher (0.827) son (0.725) learn (0.501)
Judaism: child (0.998) marriage (0.996) wife (0.995) mother (0.993) father (0.985) women (0.980) question (0.968) family (0.956) people (0.919) joseph (0.879) son (0.858) hear (0.840) learn (0.833) find (0.803) jacob (0.758) sarah (0.691) status (0.554) teach (0.533) gentile (0.521)
MOST PROMINENT FEATURES: husband [0.292] parent [0.290] nursing [0.263] spouse [0.263] elderly [0.262] unanswered [0.262] relative [0.261] grandchild [0.259] daughter [0.259] pose [0.259] answer [0.257] marry [0.256] dad [0.256] extended [0.256] pregnant [0.253] wife [0.253] sister [0.251] aunt [0.250] married [0.247] wean [0.246] adult [0.244] nurse [0.241] brother [0.241] rape [0.238] young [0.237] toy [0.235] illegitimate [0.234] mother [0.234] stranger [0.234]