Automatic Extraction of Meaning from the Web

Rudi Cilibrasi
CWI
Amsterdam, Netherlands
Email: Rudi.Cilibrasi@cwi.nl

Paul Vitanyi
CWI
Amsterdam, Netherlands
Email: Paul.Vitanyi@cwi.nl

Abstract— We consider similarity distances for two types of objects: literal objects that as such contain all of their meaning, like genomes or books, and names for objects. The latter may have literal embodiments like the first type, but may also be abstract like "red" or "christianity." For the first type we consider a family of computable distance measures corresponding to parameters expressing similarity according to particular features between pairs of literal objects. For the second type we consider similarity distances generated by web users corresponding to particular semantic relations between the (names for) the designated objects. For both families we give universal similarity distance measures, incorporating all particular distance measures in the family. In the first case the universal distance is based on compression and in the second case it is based on Google page counts related to search terms. In both cases experiments on a massive scale give evidence of the viability of the approaches.

I. INTRODUCTION

Objects can be given literally, like the literal four-letter genome of a mouse, or the literal text of *War and Peace* by Tolstoy. For simplicity we take it that all meaning of the object is represented by the literal object itself. Objects can also be given by name, like "the four-letter genome of a mouse," or "the text of *War and Peace* by Tolstoy." There are also objects that cannot be given literally, but only by name and acquire their meaning from their contexts in background common knowledge in humankind, like "home" or "red." In the literal setting, objective similarity of objects can be established by feature analysis, one type of similarity per feature. In the abstract "name" setting, all similarity must depend on background knowledge and common semantics relations, which is inherently subjective and "in the mind of the beholder."

II. COMPRESSION BASED SIMILARITY

All data are created equal but some data are more alike than others. We have recently proposed methods expressing this alikeness, using a new similarity metric based on compression. It is parameter-free in that it doesn't use any features or background knowledge about the data, and can without changes be applied to different areas and across area boundaries. It is universal in that it approximates the parameter expressing similarity of the dominant feature in all pairwise comparisons. It is robust in the sense that its success appears independent from the type of compressor used. The clustering we use is hierarchical clustering in dendrograms based on a new fast heuristic for the quartet method. The method is available as an open-source software tool, [5].

Feature-Based Similarities: We are presented with unknown data and the question is to determine the similarities among them and group like with like together. Commonly, the data are of a certain type: music files, transaction records of ATM machines, credit card applications, genomic data. In these data there are hidden relations that we would like to get out in the open. For example, from genomic data one can extract letter- or block frequencies (the blocks are over the four-letter alphabet); from music files one can extract various specific numerical features, related to pitch, rhythm, harmony etc. One can extract such features using for instance Fourier transforms [26] or wavelet transforms [13], to quantify parameters expressing similarity. The resulting vectors corresponding to the various files are then classified or clustered using existing classification software, based on various standard statistical pattern recognition classifiers [26], Bayesian classifiers [11], hidden Markov models [6], ensembles of nearest-neighbor classifiers [13] or neural networks [11], [23]. For example, in music one feature would be to look for rhythm in the sense of beats per minute. One can make a histogram where each histogram bin corresponds to a particular tempo in beats-per-minute and the associated peak shows how frequent and strong that particular periodicity was over the entire piece. In [26] we see a gradual change from a few high peaks to many low and spread-out ones going from hip-hip, rock, jazz, to classical. One can use this similarity type to try to cluster pieces in these categories. However, such a method requires specific and detailed knowledge of the problem area, since one needs to know what features to look for.

Non-Feature Similarities: Our aim is to capture, in a single similarity metric, *every effective distance*: effective versions of Hamming distance, Euclidean distance, edit distances, alignment distance, Lempel-Ziv distance, and so on. This metric should be so general that it works in every domain: music, text, literature, programs, genomes, executables, natural language determination, equally and simultaneously. It would be able to simultaneously detect *all* similarities between pieces that other effective distances can detect seperately.

Such a "universal" metric was co-developed by us in [18], [19], as a normalized version of the "information metric" of [20], [1]. Roughly speaking, two objects are deemed close if we can significantly "compress" one given the information in the other, the idea being that if two pieces are more similar, then we can more succinctly describe one given the other. The mathematics used is based on Kolmogorov complexity theory [20]. In [19] we defined a new class of (possibly non-metric) distances, taking values in $[0, 1]$ and appropriate for measuring effective similarity relations between sequences, say one type of similarity per distance, and *vice versa*. It was shown that an appropriately "normalized" information distance minorizes every distance in the class. It discovers all effective similarities in the sense that if two objects are close according to some effective similarity, then they are

also close according to the normalized information distance. Put differently, the normalized information distance represents similarity according to the dominating shared feature between the two objects being compared. In comparisons of more than two objects, different pairs may have different dominating features. For every two objects, this universal metric distance zooms in on the dominant similarity between those two objects out of a wide class of admissible similarity features. In [19] we proved its optimality and universality. The normalized information distance also satisfies the metric (in)equalities, and takes values in $[0,1]$; hence it may be called *"the" similarity metric*.

Normalized Compression Distance: Unfortunately, the universality of the normalized information distance comes at the price of noncomputability, since it is based on the uncomputable notion of Kolmogorov complexity. But since the Kolmogorov complexity of a string or file is the length of the ultimate compressed version of that file, we can use real data compression programs to approximate the Kolmogorov complexity. Therefore, to apply this ideal precise mathematical theory in real life, we have to replace the use of the noncomputable Kolmogorov complexity by an approximation using a standard real-world compressor. Thus, if C is a compressor and we use $C(x)$ to denote the length of the compressed version of a string x, then we arrive at the *Normalized Compression Distance*:

$$\text{NCD}(x,y) = \frac{C(xy) - \min(C(x), C(y))}{\max(C(x), C(y))}, \qquad (1)$$

where for convenience we have replaced the pair (x,y) in the formula by the concatenation xy, see [19], [9], In [9] we propose axioms to capture the real-world setting, and show that (1) approximates optimality. Actually, the NCD is a family of compression functions parameterized by the given data compressor C.

Universality of NCD: In [9] we prove that the NCD is universal with respect to the family of all admissible normalized distances—a special class that is argued to contain all parameters and features of similarity that are effective. The compression-based NCD method to establish a universal similarity metric (1) among objects given as finite binary strings [1], [18], [19], [9], [14], and has been applied to objects like genomes, music pieces in MIDI format, computer programs in Ruby or C, pictures in simple bitmap formats, or time sequences such as heart rhythm data, heterogenous data and anomaly detection. This method is feature-free in the sense that it doesn't analyze the files looking for particular features; rather it analyzes all features simultaneously and determines the similarity between every pair of objects according to the most dominant shared feature. The crucial point is that the method analyzes the objects themselves. This precludes comparison of abstract notions or other objects that don't lend themselves to direct analysis, like emotions, colors, Socrates, Plato, Mike Bonanno and Albert Einstein.

III. GOOGLE-BASED SIMILARITY

To make computers more intelligent one would like to represent meaning in computer-digestable form. Long-term and labor-intensive efforts like the *Cyc* project [17] and the *WordNet* project [24] try to establish semantic relations between common objects, or, more precisely, *names* for those objects. The idea is to create a semantic web of such vast proportions that rudimentary intelligence and knowledge about the real world spontaneously emerges. This comes at the great cost of designing structures capable of manipulating knowledge, and entering high quality contents in these structures by knowledgeable human experts. While the efforts are long-running and large scale, the overall information entered is minute compared to what is available on the world-wide-web.

The rise of the world-wide-web has enticed millions of users to type in trillions of characters to create billions of web pages of on average low quality contents. The sheer mass of the information available about almost every conceivable topic makes it likely that extremes will cancel and the majority or average is meaningful in a low-quality approximate sense. We devise a general method to tap the amorphous low-grade knowledge available for free on the world-wide-web, typed in by local users aiming at personal gratification of diverse objectives, and yet globally achieving what is effectively the largest semantic electronic database in the world. Moreover, this database is available for all by using any search engine that can return aggregate page-count estimates like Google for a large range of search-queries.

While the previous NCD method that compares the objects themselves using (1) is particularly suited to obtain knowledge about the similarity of objects themselves, irrespective of common beliefs about such similarities, we now develop a method that uses only the name of an object and obtains knowledge about the similarity of objects by tapping available information generated by multitudes of web users. Here we are reminded of the words of D.H. Rumsfeld [22] "A trained ape can know an awful lot/ Of what is going on in this world,/ Just by punching on his mouse/ For a relatively modest cost!" The new method is useful to extract knowledge from a given corpus of knowledge, in this case the Google database, but not to obtain true facts that are not common knowledge in that database. For example, common viewpoints on the creation myths in different religions may be extracted by the Googling method, but contentious questions of fact concerning the phylogeny of species can be better approached by using the genomes of these species, rather than by opinion.

Googling for Knowledge: Let us start with simple intuitive justification (not to be mistaken for a substitute of the underlying mathematics) of the approach we propose in [10]. The Google search engine indexes around ten billion pages on the web today. Each such page can be viewed as a set of index terms. A search for a particular index term, say "horse", returns a certain number of hits (web pages where this term occurred), say 46,700,000. The number of hits for the search term "rider" is, say, 12,200,000. It is also possible to search for the pages where both "horse" and "rider" occur. This gives, say, 2,630,000 hits. This can be easily put in the standard probabilistic framework and several similar approaches have already yielded good results such as [25].

If w is a web page and x a search term, then we write $x \in w$ to mean that Google returns web page w when presented with search term x. An *event* is a set of web pages returned by Google after it has been presented by a search term. We can view the event as the collection of all contexts of the search term, background knowledge, as induced by the accessible web

pages for the Google search engine. If the search term is x, then we denote the event by $\mathbf{x}$, and define $\mathbf{x} = \{w : x \in w\}$. The *probability* $g(x)$ of an event $\mathbf{x}$ is the number of web pages in the event divided by the sum of the number N of web pages summed over all events (counting the multiplicity of the web pages occurring in the overlap of more than one event). This number N is by definition at least the overall number M of web pages possibly returned by Google. Thus, $g(x) = |\mathbf{x}|/N \leq |\mathbf{x}|/M$. At the time of writing, Google searches $M = 8,058,044,651$ web pages. Since web pages contain not more than a certain constant of search terms, say *alpha*, we have $M \leq N \leq \alpha M$. Define the joint event $\mathbf{x} \bigcap \mathbf{y} = \{w : x,y \in w\}$ as the set of web pages returned by Google, containing both the search term x and the search term y. The joint probability $g(x,y) = |\{w : x,y \in w\}|/N$ is the number of web pages in the joint event divided by the overall number N. This notation also allows us to define the probability $g(x|y)$ of *conditional* events $\mathbf{x}|\mathbf{y} = (\mathbf{x} \bigcap \mathbf{y})/\mathbf{y}$ defined by $g(x|y) = g(x,y)/g(y)$. We can see N as a normalizing term, and often use $N = M$, the total number of web pages indexed by Google.

In the above example, with $N = M$, we have therefore $p(horse) \approx 0.0058$, $g(rider) \approx 0.0015$, $g(horse,rider) \approx 0.0003$. We conclude that the probability $g(horse|rider)$ of "horse" accompanying "rider" is $\approx 1/5$ and the probability $g(rider|horse)$ of "rider" accompanying "horse" is $\approx 1/19$. The probabilities are asymmetric, and it is the least probability that is the significant one. A very general search term like "the" occurs in virtually all (English language) web pages. Hence $g(the|rider) \approx 1$, and for almost all search terms x we have $g(the|x) \approx 1$. But $g(rider|the) \ll 1$, say about equal to $g(rider)$, and gives the relevant information about the association of the two terms.

Define the *frequency* $f(x)$ of search term x as the number of pages a Google search for x returns: $f(x) = Ng(x)$, $f(x,y) = Ng(x,y)$, and $g(x|y) = f(x,y)/f(y)$. Our formal notion, the *normalized Google distance (NGD)*, defined by

$$\text{NGD}(x,y) = \frac{\max\{\log f(x), \log f(y)\} - \log f(x,y)}{\log N - \min\{\log f(x), \log f(y)\}}, \quad (2)$$

and if $f(x), f(y) > 0$ and $f(x,y) = 0$ then $\text{NGD}(x,y) = \infty$. From (2) we see that

1) $\text{NGD}(x,y)$ is undefined for $f(x) = f(y) = 0$;
2) $\text{NGD}(x,y) = \infty$ for $f(x,y) = 0$ and either or both $f(x) > 0$ and $f(y) > 0$; and
3) $\text{NGD}(x,y) \geq 0$ otherwise.

With the Google hit numbers above, and $N = M$, we can now compute

$$\text{NGD}(horse, rider) \approx 0.443.$$

We did the same calculation when Google indexed only one-half of the current number of pages: 4,285,199,774. It is instructive that the probabilities of the used search terms didn't change significantly over this doubling of pages, with number of hits for "horse" equal 23,700,000, for "rider" equal 6,270,000, and for "horse, rider" equal to 1,180,000. The $\text{NGD}(horse, rider)$ we computed in that situation was 0.460. This is in line with our contention that the relative frequencies of web pages containing search terms gives objective information about the semantic relations between the search terms. If this is the case, then with the vastness of the information

accessed by Google, the Google probabilities of search terms, and the computed NGD 's should stabilize (be scale invariant) with a growing Google database.

IV. FROM NCD TO NGD

The Google Distribution: Let the set of singleton *Google search terms* be denoted by $\mathcal{S}$. In the sequel we use both singleton search terms and doubleton search terms $\{\{x,y\} : x,y \in \mathcal{S}\}$. Let the set of web pages indexed (possible of being returned) by Google be Ω. The cardinality of Ω is denoted by $M = |\Omega|$, and currently $8 \cdot 10^9 \leq M \leq 9 \cdot 10^9$. Assume that a priori all web pages are equi-probable, with the probability of being returned by Google being $1/M$. A subset of Ω is called an *event*. Every *search term x* usable by Google defines a *singleton Google event* $\mathbf{x} \subseteq \Omega$ of web pages that contain an occurrence of x and are returned by Google if we do a search for x. Let $L : \Omega \to [0,1]$ be the uniform mass probability function. The probability of such an event $\mathbf{x}$ is $L(\mathbf{x}) = |\mathbf{x}|/M$. Similarly, the *doubleton Google event* $\mathbf{x} \bigcap \mathbf{y} \subseteq \Omega$ is the set of web pages returned by Google if we do a search for pages containing both search term x and search term y. The probability of this event is $L(\mathbf{x} \bigcap \mathbf{y}) = |\mathbf{x} \bigcap \mathbf{y}|/M$. We can also define the other Boolean combinations: $\neg \mathbf{x} = \Omega \backslash \mathbf{x}$ and $\mathbf{x} \bigcup \mathbf{y} = \neg(\neg \mathbf{x} \bigcap \neg \mathbf{y})$, each such event having a probability equal to its cardinality divided by M. If $\mathbf{e}$ is an event obtained from the basic events $\mathbf{x}, \mathbf{y}, \ldots$, corresponding to basic search terms $x, y, \ldots$, by finitely many applications of the Boolean operations, then the probability $L(\mathbf{e}) = |\mathbf{e}|/M$.

Google events capture in a particular sense all background knowledge about the search terms concerned available (to Google) on the web. Therefore, it is natural to consider code words for those events as coding this background knowledge. However, we cannot use the probability of the events directly to determine a prefix code such as the Shannon-Fano code [20]. The reason is that the events overlap and hence the summed probability exceeds 1. By the Kraft inequality [20] this prevents a corresponding Shannon-Fano code. The solution is to normalize: We use the probability of the Google events to define a probability mass function over the set $\{\{x,y\} : x,y \in \mathcal{S}\}$ of Google search terms, both singleton and doubleton. Define

$$N = \sum_{\{x,y\} \subseteq \mathcal{S}} |\mathbf{x} \bigcap \mathbf{y}|,$$

counting each singleton set and each doubleton set (by definition unordered) once in the summation. Since every web page that is indexed by Google contains at least one occurrence of a search term, we have $N \geq M$. On the other hand, web pages contain on average not more than a certain constant α search terms. Therefore, $N \leq \alpha M$. Define

$$g(x) = g(x,x), \quad g(x,y) = L(\mathbf{x} \bigcap \mathbf{y})M/N = |\mathbf{x} \bigcap \mathbf{y}|/N. \quad (3)$$

Then, $\sum_{\{x,y\} \subseteq \mathcal{S}} g(x,y) = 1$. Note that $g(x,y)$ is not a conventional joint distribution since possibly $g(x) \neq \sum_{y \in \mathcal{S}} g(x,y)$. Rather, we consider g to be a probability mass function over the sample space $\{\{x,y\} : \{x,y\} \subseteq \mathcal{S}\}$. This g-distribution changes over time, and between different samplings from the distribution. But let us imagine that g holds in the sense of an instantaneous snapshot. The real situation will be an approximation of this. Given the Google machinery, these are

absolute probabilities which allow us to define the associated Shannon-Fano code for both the singletons and the doubletons.

Normalized Google Distance The *Google code* length G is defined by

$$G(x) = G(x,x), \quad G(x,y) = \log 1/g(x,y). \tag{4}$$

In contrast to strings x where the complexity $C(x)$ represents the length of the compressed version of x using compressor C, for a search term x (just the name for an object rather than the object itself), the Google code of length $G(x)$ represents the shortest expected prefix-code word length of the associated Google event $\mathbf{x}$. The expectation is taken over the Google distribution p. In this sense we can use the Google distribution as a compressor for Google "meaning" associated with the search terms. The associated NCD , now called the *normalized Google distance (NGD)* is then defined by (2) with N substituted for M, rewritten as

$$\text{NGD}(x,y) = \frac{G(x,y) - \min(G(x), G(y))}{\max(G(x), G(y))}. \tag{5}$$

This NGD is an approximation to the NID using the Shannon-Fano code (Google code) generated by the Google distribution as defining a compressor approximating the length of the Kolmogorov code, using the background knowledge on the web as viewed by Google as conditional information. In experimental practice, we consider N (or M) as a normalization constant that can be adjusted.

Universality of NGD: In the full paper [10] we show that (2) and (5) are close in typical situations. Our experimental results suggest that every reasonable (greater than any $f(x)$) value can be used for the normalizing factor N, and our results seem in general insensitive to this choice. In our software, this parameter N can be adjusted as appropriate, and we often use M for N. In the full paper we analyze the mathematical properties of NGD , and prove the universality of the Google distribution among web author based distributions, as well as the universality of the NGD with respect to the family of the individual web author's NGD 's, that is, their individual semantics relations, (with high probability)—not included here for space reasons.

V. APPLICATIONS

Applications of NCD: We developed the CompLearn Toolkit, [5], and performed experiments in vastly different application fields to test the quality and universality of the method. The success of the method as reported below depends strongly on the judicious use of encoding of the objects compared. Here one should use common sense on what a real world compressor can do. There are situations where our approach fails if applied in a straightforward way. For example: comparing text files by the same authors in different encodings (say, Unicode and 8-bit version) is bound to fail. For the ideal similarity metric based on Kolmogorov complexity as defined in [19] this does not matter at all, but for practical compressors used in the experiments it will be fatal. Similarly, in the music experiments below we use symbolic MIDI music file format rather than wave format music files. The reason is that the strings resulting from straightforward discretizing the wave form files may be too sensitive to how we discretize. Further research may ovecome this problem.

The NCD is not restricted to a specific application area, and works across application area boundaries. To extract a hierarchy of clusters from the distance matrix, we determine a dendrogram (binary tree) by a new quartet method and a fast heuristic to implement it. The method is implemented and available as public software [5], and is robust under choice of different compressors. This approach gives the first completely automatic construction of the phylogeny tree based on whole mitochondrial genomes, [18], [19], a completely automatic construction of a language tree for over 50 Euro-Asian languages [19], detects plagiarism in student programming assignments [4], gives phylogeny of chain letters [2], and clusters music [8]. Moreover, the method turns out to be robust under change of the underlying compressor-types: statistical (PPMZ), Lempel-Ziv based dictionary (gzip), block based (bzip2), or special purpose (Gencompress).

To substantiate our claims of universality and robustness, in [9] we report evidence of successful application in areas as diverse as genomics, virology, languages, literature, music, handwritten digits, astronomy, and combinations of objects from completely different domains, using statistical, dictionary, and block sorting compressors. In genomics we presented new evidence for major questions in Mammalian evolution, based on whole-mitochondrial genomic analysis: the Eutherian orders and the Marsupionta hypothesis against the Theria hypothesis. Apart from the experiments reported in [9], the clustering by compression method reported in this paper has recently been used in many different areas all over the world. One item in our group was to analyze network traffic and cluster computer worms and virusses [27]. Finally, recent work [14] reports experiments with our method on all time sequence data used in all the major data-mining conferences in the last decade. Comparing the compression method with all major methods used in those conferences they established clear superiority of the compression method for clustering heterogenous data, and for anomaly detection.

Applications of NGD: This new method is proposed in [10] to extract semantic knowledge from the world-wide-web for both supervised and unsupervised learning using the Google search engine in an unconventional manner. The approach is novel in its unrestricted problem domain, simplicity of implementation, and manifestly ontological underpinnings. We give evidence of elementary learning of the semantics of concepts, in contrast to most prior approaches (outside of Knowledge Representation research) that have neither the appearance nor the aim of dealing with ideas, instead using abstract symbols that remain permanently ungrounded throughout the machine learning application. The world-wide-web is the largest database on earth, and it induces a probability mass function, the Google distribution, via page counts for combinations of search queries. This distribution allows us to tap the latent semantic knowledge on the web. While in the NGD compression-based method one deals with the objects themselves, in the current work we deal with just names for the objects. In [10], as proof of principle, we give examples of:

(i) Unsupervised hierarchical clustering, demonstrating the ability to distinguish between colors and numbers, cluster names of paintings by 17th century Dutch masters and names of books by English novelists;

complearn version 0.8.19
tree score S(T) = 0.940416
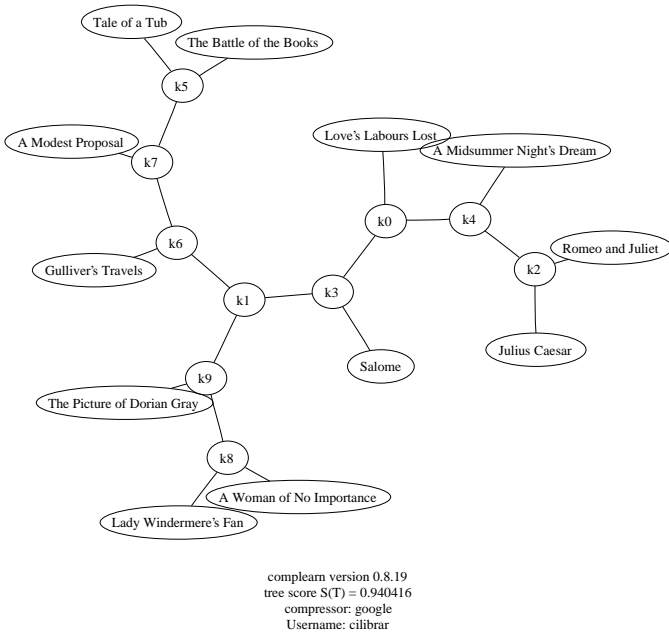compressor: google
Username: cilibrar

Fig. 1.   Hierarchical clustering of authors

(ii) Supervised concept-learning by example, using Support Vector Machines, demonstrating the ability to understand electrical terms, religious terms, emergency incidents, and by conducting a massive experiment in understanding WordNet categories [7]; and

(iii) Matching of meaning, in an example of automatic English-Spanish translation.

Because of space considerations, we present the outcome of just one experiment, Figure 1, the example of the English novelists:

William Shakespeare: *A Midsummer Night's Dream; Julius Caesar; Love's Labours Lost; Romeo and Juliet* .

Jonathan Swift: *The Battle of the Books; Gulliver's Travels; Tale of a Tub; A Modest Proposal*;

Oscar Wilde: *Lady Windermere's Fan; A Woman of No Importance; Salome; The Picture of Dorian Gray*.

The various other experiments we performed gave altogether similar results. The reader can do his own experiments to satisfy his curiosity using our publicly available software tool at http://clo.complearn.org/, also used in the depicted experiments. On the web page http://clo.complearn.org/clo/listmonths/t.html the onging cumulated results of all (in December 2005 some 160) experiments by the public, including the one depicted here, are recorded.

VI. Conclusion

It appears to us remarkable, that a method that is totally unsupervised and feature-free, that doesn't get any specific background information whatsoever, uses only Google hit counts on the world-wide-web (which by the way are notoriously untrustworthy), is able to distill this accurate semantic information. Because it is feature-free and contents-free in its invocation, but uses the vast amount of information on the world-wide-web, it will work on any domain, on heterogenous

data (rather, names for data), and will do anomaly detection in a low-quality manner.

REFERENCES

[1] C.H. Bennett, P. Gács, M. Li, P.M.B. Vitányi, W. Zurek, "Information Distance," *IEEE Trans. Information Theory*, 44:4(1998), pp. 1407–1423.
[2] C.H. Bennett, M. Li, B. Ma, "Chain letters and evolutionary histories," *Scientific American*, June 2003, pp. 76–81.
[3] C.J.C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, 2:2(1998),pp. 121–167.
[4] X. Chen, B. Francia, M. Li, B. McKinnon, A. Seker, "Shared information and program plagiarism detection," *IEEE Trans. Inform. Th.*, 50:7(2004), pp. 1545–1551.
[5] R. Cilibrasi, The CompLearn Toolkit, CWI, 2003, Available: http://www.complearn.org/
[6] W. Chai and B. Vercoe. "Folk music classification using hidden Markov models," *Proc. of International Conference on Artificial Intelligence*, 2001.
[7] R. Cilibrasi, P. Vitanyi, "Automatic Meaning Discovery Using Google: 100 Experiments in Learning WordNet Categories," 2004, Available: http://www.cwi.nl/~cilibrar/googlepaper/appendix.pdf
[8] R. Cilibrasi, R. de Wolf, P. Vitanyi. "Algorithmic clustering of music based on string compression," *Computer Music J.*, 28:4(2004), pp. 49-67.
[9] R. Cilibrasi, P.M.B. Vitanyi, "Clustering by compression," *IEEE Trans. Information Theory*, 51:4(2005), pp. 1523-1545.
[10] R. Cilibrasi, P. Vitanyi, "Automatic meaning discovery using Google," Manuscript, CWI, 2004; http://arxiv.org/abs/cs.CL/0412098
[11] R. Dannenberg, B. Thom, and D. Watson. "A machine learning approach to musical style recognition," *Proc. International Computer Music Conference*, pp. 344-347, 1997.
[12] "The basics of Google search," http://www.google.com/help/basics.html.
[13] M. Grimaldi, A. Kokaram, and P. Cunningham. "Classifying music by genre using the wavelet packet transform and a round-robin ensemble," Technical report TCD-CS-2002-64, Trinity College Dublin, 2002. Available: http://www.cs.tcd.ie/publications/tech-reports/reports.02/TCD-CS-2002-64.pdf
[14] E. Keogh, S. Lonardi, and C.A. Rtanamahatana, "Toward parameter-free data mining," In: *Proc. 10th ACM SIGKDD Intn'l Conf. Knowledge Discovery and Data Mining*, Seattle, Washington, USA, August 22—25, 2004, pp. 206–215.
[15] A.N. Kolmogorov. "Three approaches to the quantitative definition of information," *Problems Inform. Transmission*, 1:1(1965), pp. 1–7.
[16] A.N. Kolmogorov. "Combinatorial foundations of information theory and the calculus of probabilities," *Russian Math. Surveys*, 38:4(1983), pp. 29–40.
[17] D. B. Lenat. "Cyc: A large-scale investment in knowledge infrastructure," *Comm. ACM*, 38:11(1995),pp. 33–38.
[18] M. Li, J.H. Badger, X. Chen, S. Kwong, P. Kearney, and H. Zhang, "An information-based sequence distance and its application to whole mitochondrial genome phylogeny," *Bioinformatics*, 17:2(2001), pp. 149–154.
[19] M. Li, X. Chen, X. Li, B. Ma, P. Vitanyi. "The similarity metric," *IEEE Trans. Information Theory*, 50:12(2004), pp. 3250–3264.
[20] M. Li, P. M. B. Vitanyi. *An Introduction to Kolmogorov Complexity and Its Applications*, 2nd Ed., Springer-Verlag, New York, 1997.
[21] S. L. Reed, D. B. Lenat. "Mapping ontologies into cyc," *Proc. AAAI Conference 2002 Workshop on Ontologies for the Semantic Web*, Edmonton, Canada. Available: http://citeseer.nj.nec.com/509238.html
[22] D.H. Rumsfeld, "The digital revolution," originally published June 9, 2001, following a European trip. In: H. Seely, The Poetry of D.H. Rumsfeld, 2003, Available: http://slate.msn.com/id/2081042/
[23] P. Scott. "Music classification using neural networks," 2001. Available: http://www.stanford.edu/class/ee373a/musicclassification.pdf
[24] G.A. Miller et.al, WordNet, "A Lexical Database for the English Language," Cognitive Science Lab, Princeton University. Available: http://www.cogsci.princeton.edu/~wn
[25] P. D. Turney, "Mining the Web for synonyms: PMI-IR versus LSA on TOEFL," *Proc. of the Twelfth European Conference on Machine Learning (ECML2001)*, Freiburg, Germany, pp. 491–502, 2001.
[26] G. Tzanetakis and P. Cook, "Music genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, 10(5), pp. 293–302, 2002.
[27] S. Wehner, "Analyzing network traffic and worms using compression," http://arxiv.org/abs/cs.CR/0504045