

# **Introduction to the Normalized Google Distance**

Päivi Hanne-Marie Suomalainen

Helsinki 10th April 2007

Essay **\*\*DRAFT\*\***

UNIVERSITY OF HELSINKI  
Department of Computer Science

Faculty of Science

Department of Computer Science

Tekijä — Författare — Author

Päivi Hanne-Marie Suomalainen

Työn nimi — Arbetets titel — Title

Introduction to the Normalized Google Distance

Oppiaine — Läroämne — Subject

Computer Science

Työn laji — Arbetets art — Level

Essay **\*\*DRAFT\*\***

Aika — Datum — Month and year

10th April 2007

Sivumäärä — Sidoantal — Number of pages

7 pages

Tiivistelmä — Referat — Abstract

This essay introduces Rudi Cilibrasi's and Paul Vitányi's novel approach to measuring semantic similarity between words and phrases based on Google page counts and on the optimal but noncomputable similarity metric called the Normalized Information Distance, NID. The resulting similarity distance is called the Normalized Google Distance, or shortly, the NGD. We start by briefly introducing the theory behind the NGD: the Kolmogorov complexity, the information distance, the normalized information distance (NID), and the normalized compression distance (NCD). After that, we introduce the Google distribution and the NGD, certainly not forgetting the applications and the experiments (some of our own and some of Cilibrasi's and Vitányi's).

(If I have any time, I'll introduce the quartet tree heuristic also...)

NB: this is still a **\*VERY\*** early draft version, so most of the stuff is still missing... My apologies to the ones who'll have to review this...

Avainsanat — Nyckelord — Keywords

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Seminar: Language Technology and Applications, spring 2007

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Googling for Similarity</b>	<b>1</b>
2.1	Kolmogorov Complexity . . . . .	1
2.2	Information Distance . . . . .	2
2.3	Normalized Information Distance . . . . .	3
2.4	Normalized Google Distance . . . . .	4
<b>3</b>	<b>Quartet Tree Heuristic</b>	<b>6</b>
<b>4</b>	<b>Applications and Experiments</b>	<b>6</b>
4.1	Hierarchical Clustering . . . . .	6
4.2	Classification . . . . .	6
<b>5</b>	<b>Summary</b>	<b>6</b>
	<b>References</b>	<b>7</b>

# 1 Introduction

still missing... sorry.

## 2 Googling for Similarity

missing stuff...

### 2.1 Kolmogorov Complexity

Let us think about the problem of measuring the *information content* or *complexity* of an individual finite object for a while. Intuitively, the complexity of an object should be somehow related to the ease of describing the object. For example, one would probably argue that a string consisting of million ones contains little information, because it can be described in few bits: "million ones". On the other hand, one would probably consider a string of zeros and ones generated by flipping a fair coin million times rather complex, or *random*, because the shortest way to describe the string would be to write it down symbol by symbol.

A solution to the problem of measuring the amount of information in an individual object is presented in the form of Kolmogorov complexity theory [LV97]. To be precise, the variant introduced shortly is the so-called *prefix* Kolmogorov complexity, where the set of halting programs for the fixed reference universal programming system forms a prefix-code. In other words, no such program is a proper prefix of any other program.

[TODO: Kraft inequality & explain prefix-code in terms of a binary tree + a picture...]

The *Kolmogorov complexity*  $K(x)$  of string  $x$  is the length of the shortest binary program  $p$  for a fixed reference universal programming system  $U$  (such as Java, LISP or a fixed reference universal prefix Turing machine in a given standard enumeration of prefix Turing machines) that prints out  $x$  and halts:

$$K(x) = \min\{\ell(p) : U(p) = x\},$$

where  $\ell(x)$  is the length of program  $p$ . Similarly, the conditional Kolmogorov complexity  $K(x|y)$  of  $x$  given  $y$  for free is the length of the shortest binary program for  $U$  that prints out  $x$  given  $y$  as input and halts:

$$K(x|y) = \min\{\ell(p) : U(p, y) = x\},$$

and the Kolmogorov complexity  $K(x, y)$  of a pair  $x, y$  is the length of the shortest binary program for  $U$  that prints out  $x$  and  $y$  and a way to tell them apart, and halts.

Obviously, the Kolmogorov complexity of an object depends on the used programming system, which in turn implies that the measure cannot be an *objective* one. Fortunately, the complexity of an object turns out to be almost independent of the choice of the universal programming system, due to the ability of any universal programming system to imitate any other universal programming system. In other words, the choice of a programming system changes the complexity of a string by at most an additive constant independent of the string, which is an ignorable quantity when considering long strings. Due to the invariance property one might say that the Kolmogorov complexity provides a nearly or an asymptotically *absolute* measure of the amount of information in *an individual object*, in contrast to Shannon's *entropy* [Sha48], which measures the expected amount of information needed to communicate/describe *an outcome of a random source*.

One drawback of the theory is that the Kolmogorov complexity is a noncomputable function, because of the undecidability of the halting problem. In other words, there exists no such program which outputs  $K(x)$  when given string  $x$  as input. Although the Kolmogorov complexity is noncomputable, it is upper semi-computable. In other words, it can be approximated from above with increasing accuracy.

## 2.2 Information Distance

In Section 2.1 we found out that the Kolmogorov complexity is an absolute measure of information in an individual object. Similar measure of absolute information distance between individual objects is obtained via the *information distance* [BGL98, LV97]. Intuitively, the minimal information distance between two strings  $x$  and  $y$ , not necessarily of the same length, is the length, in bits, of the shortest program for an universal programming system which transforms them back and forth.

More formally, the information distance  $E$  between two binary strings  $x$  and  $y$  is defined as follows:

$$E(x, y) = \min\{\ell(p) : U(p, x) = y, U(p, y) = x\}.$$

It turns out that the distance  $E$  is actually a metric (it satisfies the metric (in)equalities up to an additive constant) and it equals (up to an additive logarithmic term) to the *max distance*:

$$E_1(x, y) = \max\{K(x|y), K(y|x)\}.$$

The information distance is also *universal* in a sense that it discovers every feature defining a distance between the two strings under consideration, and it determines the distance according to the dominating feature. In other words, if the strings are close according to some 'admissible' distance, then they are at least as close according to  $E$  (up to an additive constant depending on the admissible distance but not on the strings).

A total nonnegative real-valued function  $D$  on pairs of binary strings  $x, y$  is an *admissible distance* if it is upper semi-computable, symmetric, and for every pair of

binary strings  $x, y$  the distance  $D(x, y)$  is the length of a binary prefix codeword that is a program that computes  $x$  from  $y$  and  $y$  from  $x$  [LCL04]. Obviously,  $D$  is an admissible distance. The prefix requirement ensures that the number of objects within a given distance of an object is restricted, which indeed is a desirable feature. In other words, the following version of the so-called Kraft inequality is satisfied:

$$\sum_y 2^{-D(x,y)} \leq 1.$$

Hence, unrealistic distances, e.g.  $f(x, y) = 1/2$  for every pair of strings  $x, y$ , with  $x \neq y$ , are excluded.

Unfortunately, when the goal is to measure similarity, the information distance is not an appropriate measure, due to the fact that it measures absolute distance between a pair of strings, instead of relative one. Let us consider, for example, a pair of strings of length  $n$  having an information distance  $k$ , and another pair of strings of length  $m$ , with  $m \ll n$ , having the same information distance  $k$ . Obviously, the strings of the latter pair are much more different than the strings of the former pair, because of the length factor. The solution to the problem is normalization. The normalized version of the information distance is introduced in Section 2.3.

### 2.3 Normalized Information Distance

In order to measure similarity of objects, one is interested in relative distances, rather than absolute ones. In other words, the focus of interest lies in the normalized versions of the absolute distances, with distance 0 whenever the objects are maximally similar and distance 1 when the objects are maximally dissimilar. A total nonnegative real-valued function  $d$  on pairs of binary strings  $x, y$ , is a *normalized distance* or a *similarity distance* [LCL04] if the range of  $d$  is  $[0, 1]$ ,  $d$  is symmetric, and the following density requirement is satisfied for every constant  $e \in [0, 1]$  and for every binary string  $x$ :

$$|\{y : d(x, y) \leq e \leq 1\}| < 2^{eK(x)+1}. \quad (1)$$

If a similarity distance  $d$  satisfies a normalized version of the Kraft inequality,

$$\sum_y 2^{-d(x,y)K(x)} \leq 1,$$

then the density requirement given in Equation (1) is automatically satisfied.

The question at hand is whether there exists a similarity distance having the metric and universality features of the information distance introduced in Section 2.2. In other words, is it possible to normalize the information distance somehow without losing those features? It turns out that this is actually possible by dividing the information distance by the maximum of the unconditional Kolmogorov complexities. The resulting normalized version called the *normalized information distance*

[LCL04, Cil07], or simply the NID, is defined as follows:

$$\text{NID}(x, y) = \frac{\max\{K(x|y), K(y|x)\}}{\max\{K(x), K(y)\}}. \quad (2)$$

The NID is also a metric and universal in a sense that it minorizes every upper semi-computable similarity distance up to a negligible additive error term, while the NID itself may not even be upper semi-computable. In other words, if two binary strings are similar according to some similarity distance  $d$ , then they are at least as similar (up to an additive error term) according to the NID. Hence, the NID is sometimes referred to as *the* similarity metric.

Unfortunately, the NID is noncomputable, due to the previously mentioned fact that the function  $K$  is noncomputable. However, we can use real data compression algorithms to approximate the unconditional Kolmogorov complexities from above [Cil07]. The denominator of the NID (Equation (2)) can be approximated with  $\max\{C(x), C(y)\}$ , where  $C(x)$  is the length of the compressed version of string  $x$  using compression algorithm  $C$ . The numerator on the other hand can be rewritten as  $K(x, y) - \min\{K(x), K(y)\}$ , within logarithmic additive precision. The resulting compression-based approximation of the NID is called the *normalized compression distance* [Cil07], or simply the NCD, and it is defined as follows:

$$\text{NCD}(x, y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}}, \quad (3)$$

where the pair  $(x, y)$  has been replaced for convenience by the concatenation  $xy$ . The NCD based on a 'normal compressor' is a similarity distance, it satisfies the metric (in)equalities, and is quasi-universal. All standard compression algorithms, e.g. gzip, bzip2, PPMZ, are argued to be normal.

## 2.4 Normalized Google Distance

Although we found out in Section 2.3 that it is possible to approximate the optimal NID with data compression algorithms and the real world approximation has actually proven to be quite a successful approach in a variety of different domains including e.g. genomics, virology, languages, literature, handwritten digits, stemmatology and music [Cil07], the straightforward approach in case of semantic relations of words and phrases is bound to fail. Put differently, the data compression algorithms ignore the meaning of an object completely, they only try to find some repetitions and biases in the data in order to achieve compression. The compression algorithms cannot even achieve compression for 'simple' meaningful strings like the digits of the transcendental number  $\pi = 3.1415\dots$  containing a constant amount of information [CV04] (there exists a short program that produces the consecutive digits of  $\pi$  forever).

So, instead of directly compressing the terms and using the NCD in Equation (3) to measure their semantic similarity, Cilibrasi and Vitányi [CV04, CV07] try to take

as much background knowledge as possible into account when approximating the NID. The background knowledge comes in on the discussion in the form of Google events. A *Google event*  $\mathbf{x}$  is the set of web pages returned by Google containing the search term  $x$ . Similarly, the joint event  $\mathbf{x} \cap \mathbf{y}$  is the set of web pages returned by Google containing the search term  $x$  as well as the search term  $y$ . Assuming a priori all web pages equally likely to be returned by Google, we can define probabilities for the events as follows:  $R(x) = |\mathbf{x}|/M$  and  $R(x, y) = |\mathbf{x} \cap \mathbf{y}|/M$ , where  $M$  is the number of pages Google indexes.

In order to connect the probabilistic framework to the NID, the authors make good use of the known fact that probabilities 'are' code-word lengths and vice versa. Given a probability distribution  $P$  over the source words, there exists a uniquely decodable code with code-word lengths

$$L(x) = \lceil \log_2 1/p(x) \rceil,$$

where  $p$  is the probability mass function. The expected code-word length of such code equals the entropy

$$H(X) = \sum_x p(x) \log_2 1/p(x)$$

of a random variable  $X \sim P$  up to one bit, which is optimal according to the noiseless coding theorem of Shannon [Sha48]. Conversely, any uniquely decodable code defines a probability distribution  $Q$  with

$$q(x) \propto 2^{-L(x)}.$$

If the code is complete, then the probabilities are given by  $2^{-L(x)}$ . One should note that this is a mere correspondence, we do not argue that the source words are actually distributed according to  $Q$ . But if this were the case, the code-word lengths would be the optimal ones in case of a complete code. We can also substitute 'prefix code' for 'uniquely decodable code' due to the Kraft inequality.

However, we must normalize the probabilities of the Google events before we can even think about determining a prefix code for the Google events, because the sum of the probabilities exceeds unity. The resulting *Google distribution* [CV07] over the set of search terms  $\{\{x, y\} : x, y \in \mathcal{S}\}$  is defined as follows:

$$g(x) = g(x, x), \quad g(x, y) = R(\mathbf{x} \cap \mathbf{y})M/N = |\mathbf{x} \cap \mathbf{y}|/N,$$

where

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

Each singleton set and each doubleton set is counted once in the summation. Now that the probabilities are normalized, we are finally able to define a prefix code for the events. The Google code-word lengths based on the Google distribution are defined as follows:

$$G(x) = G(x, x), \quad G(x, y) = \log_2 1/g(x, y).$$

In other words, the Google code-word length  $G(x)$  is the shortest expected prefix-code word length of the Google event  $\mathbf{x}$ . Using these code-word lengths we can approximate the NID with the NCD using the Google distribution as a compressor for the Google semantics. The resulting Google-based approximation of the NID called the *normalized Google distance* [CV04, CV07], or simply the NGD, is defined as follows:

$$\begin{aligned} NGD(x, y) &= \frac{G(x, y) - \min\{G(x), G(y)\}}{\max\{G(x), G(y)\}} \\ &= \frac{\max\{\log_2 f(x), \log_2 f(y)\} - \log_2 f(x, y)}{\log_2 N - \min\{\log_2 f(x), \log_2 f(y)\}}, \end{aligned} \quad (4)$$

where  $f(x)$  is the number of web pages containing the search term  $x$  and  $f(x, y)$  the number of web pages containing the search term  $x$  as well as the search term  $y$  according to Google.

... a lot of stuff still missing, such as some properties of NGD, universality of GD & NGD, ...

### 3 Quartet Tree Heuristic

## 4 Applications and Experiments

### 4.1 Hierarchical Clustering

- Finnish musicians
- maybe Finnish paintings
- maybe the numbers vs. colors of Cilibrasi & Vitanyi
- or maybe something else

### 4.2 Classification

- primes & WordNet comparisons of C&V

## 5 Summary

missing as well...

## References

- BGL98 Bennett, C. H., Gács, P., Li, M., Vitányi, P. M. B. Zurek, W. H., Information Distance. *IEEE Transactions on Information Theory*, 44,4(1998), 1407–1423.
- Cil07 Cilibrasi, R., *Statistical Inference Through Data Compression.* , University of Amsterdam, Holland, 2007. URL <http://cilibrar.com/projsup/thesis.pdf>.
- CV04 Cilibrasi, R. Vitányi, P. M. B., Automatic Meaning Discovery Using Google, December 2004. URL <http://xxx.lanl.gov/abs/cs.CL/0412098>.
- CV07 Cilibrasi, R. L. Vitányi, P. M., The Google Similarity Distance. *IEEE Transactions on Knowledge and Data Engineering*, 19,3(2007), 370–383.
- LCL04 Li, M., Chen, X., Li, X., Ma, B. Vitányi, P. M. B., The Similarity Metric. *IEEE Transactions on Information Theory*, 50,12(2004), 3250–3264.
- LV97 Li, M. Vitányi, P. M. B., *An Introduction to Kolmogorov Complexity and Its Applications.* Springer-Verlag, New York, , 1997.
- Sha48 Shannon, C. E., A Mathematical Theory of Communication. *The Bell System Technical Journal*, 27, 379–423, 623–656. URL <http://cm.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf>.