ISSN 1870-4069

Statistically Based Authorship Identification

Angel Kuri Morales

Instituto Tecnológico Autónomo de México,
Mexico

akuri@itam.mx

Abstract. Presently very large volumes of information are being regularly produced in the world. Most of this information is unstructured, lacking the properties usually expected from relational databases. One interesting issue in computer science is how we may achieve data mining on such unstructured data. Intuitively, its analysis has been attempted by devising schemes to identify patterns and trends through means such as statistical pattern learning. The basic problem of this approach is that the user has to decide, a priori, the model of the patterns and, furthermore, the way in which they are to be found in the data. This is true regardless of the kind of data, be it textual, musical, financial or otherwise. In this paper we explore an alternative paradigm in which raw data is categorized by analyzing a large corpus from which a set of categories and the different instances in each category are determined, resulting in a structured database. Then each of the instances is mapped into a numerical value which preserves the underlying patterns. This is done using CESAMO, a statistical algorithm. Every categorical instance is then replaced by the adequate numerical code. The resulting numerical database may be analyzed with the usual clustering algorithms. In this work we exemplify with a textual database and apply our method to characterize texts by different authors and present experimental evidence that the resulting databases yield clustering results which permit authorship identification from raw textual data.

Keywords. Databases, encoding, statistics, optimization.

1 Introduction

The problem of analyzing large sets of unstructured data sets has grown in importance over the last years. As of June 2019, there were more than 1,690,000 web sites in the world [1]. Due to corrections to Metcalfe's law [2] (which states that the value of a telecommunications network is proportional to *n* *(*$\log_2 n$*)* of the number of connected users of the system) there is a network world value of O(35e+08). The associated amount of data generated may be inferred from this number and even conservative metrics yield very large estimates. Most of these data are unstructured and recent commercial [3] approaches to the problem attest to the increasing importance assigned to this fact.

In the computer science community, data mining of texts [4], music [5] and general information [6, 7] is being approached with growing interest. In the vast majority of the cases, information extraction is highlighted and emphasizes the use of anaphora: the use of an expression the interpretation of which depends upon another expression in context. This approach, although intuitive and natural, suffers from the obvious disadvantage of being case-based. That is, a method developed for, say, texts in English will not be directly applicable to other languages and much less to other kinds of information. For example, even when limiting our range of study to texts stemming from corporate finance (i.e. mining industry literature for business intelligence), "horizontal" test mining (i.e. patent analysis) or life sciences research (i.e. mining biological pathway information) every one of the lines just mentioned relies on a case-by-case determination of the anaphoric usage.

The problem at the very heart of this issue is the fact that we must preserve the patterns underlying the information and it had not been treated with success in the past. In [8], however, a method to encode non-numerical data in mixed (numerical and categorical) databases was shown to be effective in preserving the embedded patterns. To do this we must consider the fact that there is a limited subset of codes which will preserve the patterns in those databases consisting of both numerical and non-numerical data (i.e. *mixed* databases or MD). To identify such pattern-preserving codes (PPC) we appeal to a statistical methodology. It is possible to statistically identify a set of PPCs by selectively sampling a bounded number of codes (corresponding to the different instances of the CAs) and demanding the method to set the size of the sample dynamically. Two issues have to be considered for this method to be defined in practice: a) How to set the size of the sample and 2) How to define the adequateness of the codes.

A note is in order: the PPCs are NOT to be assumed as an instance applicable to DBs other than the original one. That is to say: a set of PPCs (say PPC1) obtained from a DB (say DB1) is not applicable to a different DB (say DB2) even if DB1 and DB2 are structurally identical. In other words, PPC1 ≠ PPC2 for the same DB when the tuples of such DB are different.

Consider a set of *n*-dimensional tuples (say *U*) whose cardinality is *m*. Assume there are *n* unknown functions of *n-1* variables each, which we denote with $f_k(v_1,...,v_{k-1},v_{k+1},...,v_n)$;k=*1,...,n*.

Let us also assume that there is a method which allows us to approximate f_k (from the tuples) with F_k. Denote the resulting *n* functions of *n-1* independent variables with F_i, thus:

$$F_k \approx f(v_1,...,v_{k-1},v_{k+1},...,v_n);\ k=1,...,n\ . \qquad (1)$$

The difference between f_k and F_k will be denoted with ε_k such that, for attribute *k* and the *m* tuples in the database:

$$\varepsilon k = max\,[\,abs(\,fki - Fki)]\,;\ i=1,...,m\ . \qquad (2)$$

Our contention is that the PPCs are the ones which minimize ε_k for all *k*.

This is so because only those codes which retain the relationships between variable *k* and the remaining *n-1* variables AND do this for ALL variables in the ensemble will preserve the whole set of relations (i.e. patterns) present in the database, as in (3):

$$\Xi = min[\, max\, (\varepsilon k\, ;\, k=1,...,n)]\, . \qquad (3)$$

Notice that this is a multi-objective optimization problem because complying with condition *k* 0in (2) for any given value of k may induce the non-compliance for a different possible *k*. Using the min-max expression of (3) equates to selecting a particular point in Pareto's front [9]. To achieve the purported goal we must have a tool which is capable of identifying the F_k's in (1) and the codes which attain the minimization of (3).

For this purpose we designed a new algorithm (called "CESAMO": Categorical Encoding by Statistical Applied Modeling) which relies on statistical and numerical considerations.

1.1 CESAMO Algorithm

In what follows we denote the number of tuples in the DB by *t* and the number of categorical attributes by *c*; the number of numerical attributes by *n*; the *i*-th categorical variable by *vi*; the value obtained for variable *i* as a function of variable *j* by *yi(j)*.

We will sample the codes yielding *yi* as a function of a sought for relationship. This relationship and the model of the population it implies will be selected so as to preserve the behavioral patterns embedded in the DB.

Two issues are of primordial importance in the proposed methodology:

a) How to determine the number of codes to sample.

b) How to define the function which will preserve the patterns.

Regarding (b), we use a mathematical model considering high order relations. Regarding (a), we know that the distribution of the means of the samples of *yi* (yi_{AVG}) will become Gaussian. Once it becomes Gaussian we know that further sampling of the *yi*'s will not significantly modify the characterization of the population.

The general algorithm for CESAMO is as follows:

- Specify the mixed database MD.
- Specify the sample size (*ss*).
- MD is analyzed to determine *n*, *t* and *ci(i)* for *i=1,…,c*.
- The numerical data are assumed to have been mapped into [0,1). Therefore, every *ci* will be, likewise, in [0,1).

```
for $i \leftarrow 1$ to $c$
  Do until the distribution of $yi_{AVG}$ is Gaussian
     Randomly select variable $j\ (j \neq i)$
     Assign random values to all instances of $vi$.
     $yi_{AVG} \leftarrow 0$
     For k ←1 to ss
        $yi \leftarrow f(vj)$
        $yi_{AVG} \leftarrow yi_{AVG} + yi$
     endfor
```

```
10.         $yi_{AVG} = yi_{AVG}/ss$
11.     enddo
12.     Select the codes corresponding to the best value of $yi$
13.  endfor
```

Hence, we sample enough codes to guarantee the statistical stability of the values calculated from $yi \leftarrow f(vj)$. The codes corresponding to the best approximation will be those inserted in MD. CESAMO relies on a double level sampling: only pairs of variables are considered and every pair is, in itself, sampling the multivariate space. This avoids the need to explicitly solve the multi-objective optimization underlying problem. The clustering problem may be, then, numerically tackled.

Notice that *vj* may be, itself, categorical. In that cases every categorical instance of *vj* is replaced by random codes so that we may calculate *f(vj)*.

One of the key points of CESAMO is how to define the functional relation specified in step 7 (i.e. $yi \leftarrow f(vj)$). This selection defines the way in which our intent of preserving the patterns in the data is understood. In our experiments we set $yi \leftarrow P_{11}(x) \sim \beta_0 + \sum_{i=1}^{6} \beta_i x^{2i-1}$ as a universal polynomial approximation. In [10] it was shown that any continuous function may be approximately realized with a linear combination of monomials which has a constant plus terms of odd degree. In the case of $P_{11}(x)$ the relationships are not limited *a priori*. The β_i of $P_{11}(x)$ were found with the so-called *Ascent Algorithm* (AA) [11]. The codes obtained from AA are called *functional* codes.

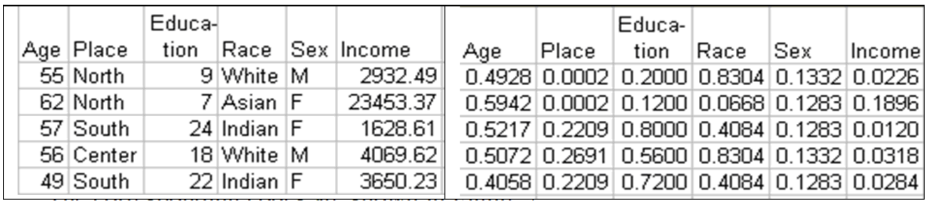
An example of a Mixed Database and the resulting Numerical Database after CESAMO is shown in Figure 1. All numerical values have been mapped into [0,1]. The corresponding codes are shown in Figure 2.

Once having shown that CESAMO does find the correct codes for categorical attributes in mixed databases any classic numerical clustering algorithm (such as Fuzzy C-Means [12] or SOMS [13]) may be used and, furthermore, any text (indeed, any collection of tokenizable data) may be treated as a set of categorical variables provided categories and their corresponding instances are identified.

The rest of the paper is organized as follows. In section 2 we describe the method we applied to tokenize English texts. In section 3 we describe the process of encoding the tokenized database to obtain the corresponding clusters. In section 4 we present an algorithm developed to identify the correspondence of the clusters in a labeled database. In section 5 we describe the experiments we performed to test the viability of our method. In section 6 we present our conclusions.

2 Tokenizing English Texts

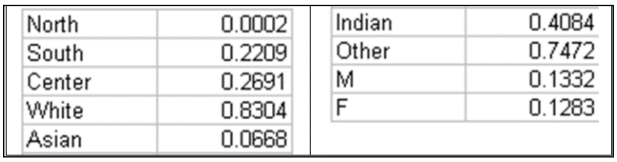
To tokenize unstructured data, we first have to find a representative sample which adequately characterizes the universe of data (*U*). Once having done so we select the number of categories (*c*). The next step is to determine the instances within every category (*t*). The number of tokens (*k*) per tuple determines the form of the structured database. The final step is to select the text to tokenize and identify the tokens which



Age	Place	Educa-tion	Race	Sex	Income
55	North	9	White	M	2932.49
62	North	7	Asian	F	23453.37
57	South	24	Indian	F	1628.61
56	Center	18	White	M	4069.62
49	South	22	Indian	F	3650.23

Age	Place	Educa-tion	Race	Sex	Income
0.4928	0.0002	0.2000	0.8304	0.1332	0.0226
0.5942	0.0002	0.1200	0.0668	0.1283	0.1896
0.5217	0.2209	0.8000	0.4084	0.1283	0.0120
0.5072	0.2691	0.5600	0.8304	0.1332	0.0318
0.4058	0.2209	0.7200	0.4084	0.1283	0.0284

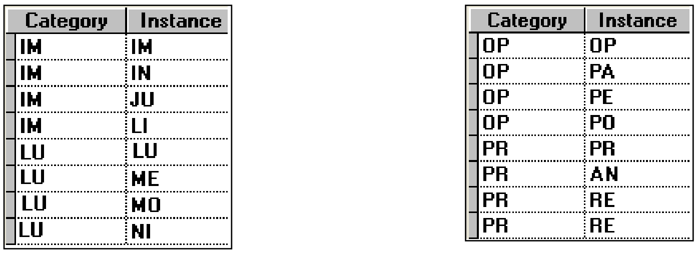
Fig. 1. Example of Mixed and Numerical Data after CESAMO.



North	0.0002
South	0.2209
Center	0.2691
White	0.8304
Asian	0.0668

Indian	0.4084
Other	0.7472
M	0.1332
F	0.1283

Fig. 2. Code for categorical instances.



Category	Instance
IM	IM
IM	IN
IM	JU
IM	LI
LU	LU
LU	ME
LU	MO
LU	NI

Category	Instance
OP	OP
OP	PA
OP	PE
OP	PO
PR	PR
PR	AN
PR	RE
PR	RE

Fig. 3. An example of categories and instances.

will populate the structured database. Notice that these steps are independent of the nature of *U*.

2.1 Tokenizing the Universe of Data

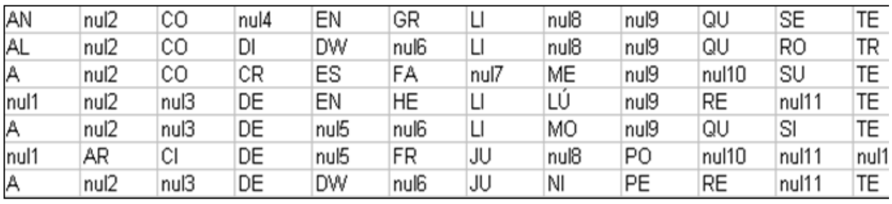
We started by selecting a collection of English texts from different authors: James Joyce, Carl Sagan, Lord Byron, William Shakespeare and English translations of Gabriel García Márquez and Julio Cortázar. 125,882 words were extracted, from which 15,031 distinct ones were identified (i.e. | *U* | = 15,301). We then made *c*=12 and *t*=4.

This meant that the words in *U* were separated into 48 intervals. Every interval consists of ≈313 words which were ordered alphabetically. A category, therefore, has 1,252 different words. Categories "IM" and "LU"; "OP" and "PR" are illustrated in Figure 3.

We say that a word starting with an instance in the (now) structured database is a token for the category. For example, the English word "*and*" is a token in category "PR". Under this classification, the sentence "*I will take a bus at twelve and arrive later tonight*" becomes the set of tokens <TE> <EN> <AU> <A> <JU> <DW> <VI> <LI> <OP> <JU> <SU>. A final specification consists of assigning a number of tokens to every tuple in the structured database. We set *k*=10. Hence, a "sentence" (which we will

Table 1. Structure of a zentence.

Zentence No.	Token No.									
1	1	2	3	4	5	6	7	8	9	10
2	1	2	3	4	5	6	7	8	9	10



AN	nul2	CO	nul4	EN	GR	LI	nul8	nul9	QU	SE	TE
AL	nul2	CO	DI	DW	nul6	LI	nul8	nul9	QU	RO	TR
A	nul2	CO	CR	ES	FA	nul7	ME	nul9	nul10	SU	TE
nul1	nul2	nul3	DE	EN	HE	LI	LÚ	nul9	RE	nul11	TE
A	nul2	nul3	DE	nul5	nul6	LI	MO	nul9	QU	SI	TE
nul1	AR	CI	DE	nul5	FR	JU	nul8	PO	nul10	nul11	nul1
A	nul2	nul3	DE	DW	nul6	JU	NI	PE	RE	nul11	TE

Fig. 4. A segment of a tokenized database.

call "zentence") has a fixed size of 10 tokens. A zentence has the structure illustrated in Table 1.

2.2 Obtaining the Tokenized Database

Not all zentences include instances of all categories and, therefore, a special token denoted by "nul$_i$" may appear in the *i-th* position of the zentence. That is, if no representative of category 1 appears in the zentence, it will be filled-in with nul1; if no representative of category 2 appears it will be filled-in with nul2 and so on. In other words, there are *(c+1)t* symbols present in the database. In our case, therefore, there are up to 60 different categorical instances present in the database. This is illustrated in Figure 4.

Once *U* is determined and categorized any given selected text in English may be mapped into a relational database formed of zentences. We selected three texts by James Joyce and three by Carl Sagan and tokenized them accordingly. These texts were then encoded by CESAMO and finally clustered using Self-Organizing Maps. The resulting clusters were labeled and then tested for similarity. Clusters whose tuples are similarly labeled indicate the same author, different authors otherwise.

3 Coding and Clustering the Tokenized Database

Once the tokenized database has been obtained, we proceed to encode the attributes, which correspond to the tokens determined as above. The tokenized database is one in which all attributes are categorical. We find a set of codes (one for each different instance of all categories) such that the structures present in the non-numerical data set are preserved when every instance of every category is replaced by its numerical counterpart. CESAMO is based on the premise that patterns are statistically preserved once the distribution of the average approximation error after encoding is Gaussian. Let us assume that we have a hypothetical *perfect* set of pattern preserving codes. Assume, also, that there are *n* attributes and *p* tuples. Given such *perfect* set it would be, in

Age	Place of Birth	Years of Study	Race	Sex	Salary
55	North	9	White	M	2,932.49
62	North	7	Asian	F	23,453.37
57	South	24	Indian	F	1,628.61
56	Center	18	White	M	4,069.62
49	South	22	Indian	F	3,650.23

Fig. 5a. A mixed database (MD).

North	Center	South	White	Asian	Indian	Other	M	F
0.0002	0.2691	0.2209	0.8304	0.0668	0.4084	0.7472	0.1332	0.1283

Fig 5b. Instances of MD and possible codes.

Age	Place of Birth	Years of Study	Race	Sex	Salary
0.4928	0.0002	0.2000	0.8304	0.1332	0.0226
0.5942	0.0002	0.1200	0.0668	0.1283	0.1896
0.5217	0.2209	0.8000	0.4084	0.1283	0.0120
0.5072	0.2691	0.5600	0.8304	0.1332	0.0318
0.4058	0.2209	0.7200	0.4084	0.1283	0.0284
0.0870	0.0002	0.8400	0.0668	0.1332	0.2306

Fig 5c. Numerical database (ND) with codes from Figure 2b.

principle, possible to express attribute *i* as a function of the remaining *n-1* with high accuracy since this *perfect* code set will lend itself to a close approximation.

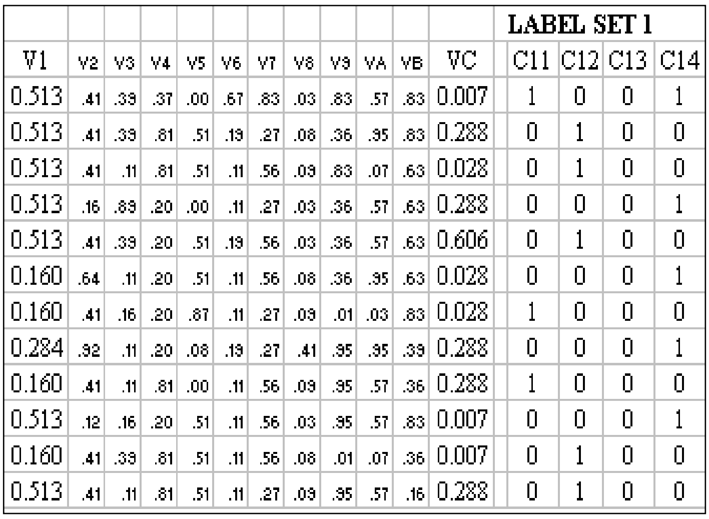
Every tuple in the DB consists of a set of possible codes which are to be assigned to every instance in the database. An encoded tuple for the database illustrated in Figure 5a is shown in Figure 5b. Figure 5c illustrates the database resulting from replacing the instances of MD with the codes of Figure 5b. Numerical variables are mapped into [0, 1] so that all numbers in the DB lie in the same range. This strategy guarantees that the resulting set of codes corresponds to the best global behavior.

That is, the final set of codes encompasses the best combinations of the f_i's minimizing the approximation error and the multi-objective optimization is solved Since the codes are arrived at from a stochastic process any two runs of CESAMO will result in different sets of codes, say S1 and S2.

This fact allows us to verify that, as postulated, patterns will be preserved. This is done by applying a clustering algorithm which yields an assignment of every tuple to one of *m* clusters. Under the assumption of pattern preservation, clustering with S1 and S2 should yield analogous clusters. This is, indeed, the fact, as was shown in [8].

4 Identification of Cluster Matching in Labeled Databases

The correct identification of analogous clusters is compulsory if, as intended, we are to determine whether two texts correspond (or not) to the same author. Texts T1A and T2A both authored by A should correspond to similar clusters whereas texts T1A and



												LABEL SET 1			
V1	V2	V3	V4	V5	V6	V7	V8	V9	VA	VB	VC	C11	C12	C13	C14
0.513	.41	.39	.37	.00	.67	.83	.03	.83	.57	.83	0.007	1	0	0	1
0.513	.41	.39	.81	.51	.19	.27	.08	.36	.95	.83	0.288	0	1	0	0
0.513	.41	.11	.81	.51	.11	.56	.09	.83	.07	.63	0.028	0	1	0	0
0.513	.16	.89	.20	.00	.11	.27	.03	.36	.57	.63	0.288	0	0	0	1
0.513	.41	.39	.20	.51	.19	.56	.03	.36	.57	.63	0.606	0	1	0	0
0.160	.64	.11	.20	.51	.11	.56	.08	.36	.95	.63	0.028	0	0	0	1
0.160	.41	.16	.20	.87	.11	.27	.09	.01	.03	.83	0.028	1	0	0	0
0.284	.92	.11	.20	.08	.19	.27	.41	.95	.95	.39	0.288	0	0	0	1
0.160	.41	.11	.81	.00	.11	.56	.09	.95	.57	.36	0.288	1	0	0	0
0.513	.12	.16	.20	.51	.11	.56	.03	.95	.57	.83	0.007	0	0	0	1
0.160	.41	.39	.81	.51	.11	.56	.08	.01	.07	.36	0.007	0	1	0	0
0.513	.41	.11	.81	.51	.11	.27	.09	.95	.57	.16	0.288	0	1	0	0

Fig. 6a. A segment of labeled Numerical Database T1.

T1B (authored, respectively, by A and B) should correspond to different clusters. To test the purported cluster similarity poses the technical problem whose solution we describe in what follows.

4.1 The Problem of Cluster Matching

Assume that tables T1 and T2 consisting of attributes V1, ..., VC have been classified into 4 clusters and labeled as illustrated in Figures 6a and 6b. This labeling convention is convenient since one may easily count the number of matches between two clusters. However, clustering algorithms do not necessarily yield the same order of the label columns.

For example, in Figure 7 we have compared column C11 to C21, on the one hand and C11 to C24 on the other. The first choice yields a count of 6 matches leading us to the conclusion that sets C11 and C21 do not match and that, therefore, T1 and T2 do not share the same clusters.

The second choice, however, yields the full 12 matches.

Therefore, in this instance one must conclude that column C11 (from set 1) actually corresponds to column C24 (from set 2). Accordingly, we should also conclude that T1 and T2 correspond to the same set for cluster 1. The correct pairing has to be achieved in similar fashion for all clusters.

If there are *m* clusters and *p* tuples there are m^p possible combinations of valid labeling sets. We need to investigate which of these does actually correspond to the proper matching of the *m* clusters in T1 with those of T2.

Only then we may compare T1 and T2 and determine their similarity. To achieve this identification, we designed the following algorithm.

4.2 Algorithm for Optimization of Cluster Matching

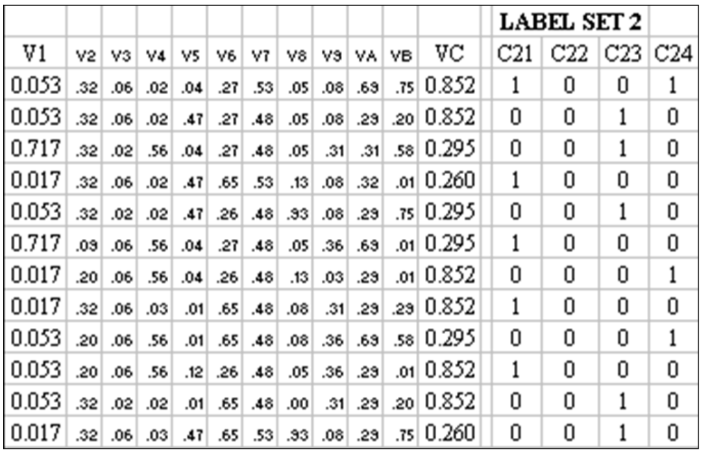
The proposed algorithm for optimization of cluster matching is as follows.

```
1. Create a matching table "MT" of dimensions $m \times m$.
   Make MT($i,j$) $\leftarrow$ 0 for all $i, j$.
2. For $i \leftarrow 1$ to $m$
      For $j \leftarrow 1$ to $m$
         If column $i$ = column $j$
            MT($i,j$) $\leftarrow$ $MT(i,j) +1$
         endif
      endfor
   endfor
  MT($i,j$) will contain the number of matches between cluster $i$ of table T1 and cluster
  $j$ of table T2.
3. Create a table "Scores" of dimension $Q$ ($Q$ >>0).
4. For $i \leftarrow$ 1 to $Q$
     4.1. Set a random valid sequence $S_i$ of $m$ possible matching sequences between
          the clusters of T1 and those of T2.
     4.2. Find the number of matches $M_i$ between T1 and T2 from table MT as per $S_i$.
     4.3. Make Scores(i) $\leftarrow$ $M_i$.
   endfor
5. $I \leftarrow$ index of max(Scores($i$)) for all $i$.
6. Select $S_I$. This is the matching set which maximizes the number of coincidences
   between the clusters of T1 and T2.
```

The core of the algorithm lies in step 4.1 where the valid matching sequences are determined. This algorithm will find the sequence which maximizes the number of matches between the clusters of T1 and T2 with high probability provided Q is large enough. In our experiments we made Q =1000. Given the large number of possible pairings between the clusters of T1 and T2 the algorithm is a practical way to select which cluster of T1 should be paired with which cluster of T2.

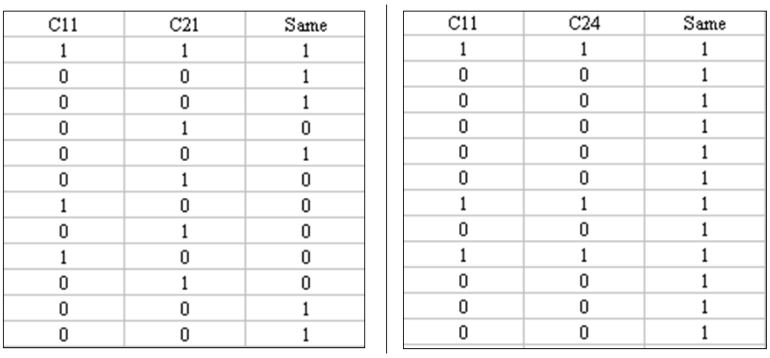
5 Experimental Authorship Identification

At this point we are in a position which allows us to test the initial hypothesis of authorship identification. As already stated, we selected 3 texts from James Joyce (JJ) and 3 from Carl Sagan (CS). These were, consecutively, tokenized, CESAMO-encoded, clustered with SOMs and labeled. Previous analysis led us to the conclusion that there were 4 clusters, as may be seen from the graph in Figure where we display the results of having trained texts from JJ and CS for up to 6 clusters with SOMs.



												LABEL SET 2			
V1	V2	V3	V4	V5	V6	V7	V8	V9	VA	VB	VC	C21	C22	C23	C24
0.053	.32	.06	.02	.04	.27	.53	.05	.08	.69	.75	0.852	1	0	0	1
0.053	.32	.06	.02	.47	.27	.48	.05	.08	.29	.20	0.852	0	0	1	0
0.717	.32	.02	.56	.04	.27	.48	.05	.31	.31	.58	0.295	0	0	1	0
0.017	.32	.06	.02	.47	.65	.53	.13	.08	.32	.01	0.260	1	0	0	0
0.053	.32	.02	.02	.47	.26	.48	.93	.08	.29	.75	0.295	0	0	1	0
0.717	.09	.06	.56	.04	.27	.48	.05	.36	.69	.01	0.295	1	0	0	0
0.017	.20	.06	.56	.04	.26	.48	.13	.03	.29	.01	0.852	0	0	0	1
0.017	.32	.06	.03	.01	.65	.48	.08	.31	.29	.29	0.852	1	0	0	0
0.053	.20	.06	.56	.01	.65	.48	.08	.36	.69	.58	0.295	0	0	0	1
0.053	.20	.06	.56	.12	.26	.48	.05	.36	.29	.01	0.852	1	0	0	0
0.053	.32	.02	.02	.01	.65	.48	.00	.31	.29	.20	0.852	0	0	1	0
0.017	.32	.06	.03	.47	.65	.53	.93	.08	.29	.75	0.260	0	0	1	0

Fig. 6b. A segment of labeled Numerical Database T2.



C11	C21	Same
1	1	1
0	0	1
0	0	1
0	1	0
0	0	1
0	1	0
1	0	0
0	1	0
1	0	0
0	1	0
0	0	1
0	0	1

C11	C24	Same
1	1	1
0	0	1
0	0	1
0	0	1
0	0	1
0	0	1
1	1	1
0	0	1
1	1	1
0	0	1
0	0	1
0	0	1

Fig. 7. Similarity for different choices of cluster columns.

The maximum errors relative to the mean were smallest for 4 clusters and the calculated standard deviation with a 0.05 *p-value* [14] was, likewise, smallest for the same number. The texts we selected were roughly the same size (≈16,000 words). They were then tokenized and CESAMO-encoded. The resulting databases were clustered and labeled. Next cluster matching was performed and adjusted when needed. We then proceeded to obtain a matrix of coincidences for the 15 possible combinations of the 6 texts. These are shown in Table 2. JJ_i denotes the *i-th* text by James Joyce; likewise CS_i denotes the texts by Sagan.

The texts pairings were ordered according to the percentage of matches we obtained. We observed the following behavior:

- All matching text matches were higher when the authors were the same, with the exception of item 6, where the texts by JJ vs CS had higher matches than expected.

- The correct assessment of authorship matches for the first 5 couples remains very close or above 75%. Therefore, a matching percentage of 75% appears to be sufficient to ascertain similar authorship.

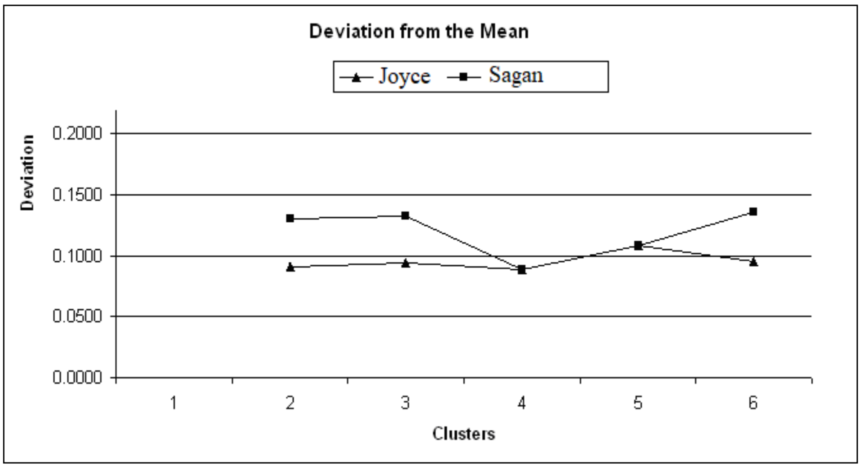


Fig. 8. Standard deviation for training errors.

Table 2. Comparison of clusters obtained.

	Text 1	**Text 2**	**Cluster Matches %**
1	CS1	CS2	98.23 %
2	CS2	CS3	86.43 %
3	CS1	CS3	82.50 %
4	JJ1	JJ2	78.54 %
5	JJ1	JJ3	74.69 %
6	JJ3	CS3	68.60 %
7	JJ2	JJ3	66.77 %
8	JJ1	CS3	65.71 %
9	JJ2	CS1	57.14 %
10	JJ3	CS2	57.14 %
11	JJ1	CS2	54.29 %
12	JJ2	CS3	54.29 %
13	JJ3	CS1	54.29 %
14	JJ1	CS1	51.52%
15	JJ2	CS2	48.57 %

- There appears to be no possible definitive conclusions of the purported authorship in the borderline percentages for items 6-8.

- Matching percentages below 60% seem to imply negative authorship for the analyzed couples.
- The identification percentage falls smoothly so that there is not a clear cut threshold dividing correctly assessed authorship from the alternative.

6 Conclusions

We have described a method to identify the authorship of selected English texts which is not based on linguistic considerations. It relies on the identification and preservation of the patterns embedded in the texts by the intelligent encoding of the data. There are several parameters which were heuristically determined and have to be further explored: e.g. the size of the zentences, the number of categories and the corresponding instances, the selected texts and their lengths. Setting them after systematic experimental tests might improve the algorithm significantly. Finally, the results, which seem to be promising, are only valid if we assume that the method will behave similarly if the restricted number of authors we selected were to be expanded. Experimental work remains to be done.

At any rate this seems to be a promising and novel alternative; particularly in view of the fact that, as pointed out in the introduction, it may be applied to any kind of unstructured data. We expect to report on the application of our method to more general and non-textual data in the near future.

References

1. Internetlivestats: https://www.internetlivestats.com/total-number-of-websites/ (2019)
2. Odlyzko, A., Tilly, B.: A refutation of Metcalfe's Law and a better estimate for the value of networks and network interconnections. Manuscript (2005)
3. IBM: http://www-03.ibm.com/press/us/en/pressrelease/46205.wss (2015)
4. Tan, A.H.: Text mining: The state of the art and the challenges. In: Proceedings of the PAKDD, Workshop on Knowledge Discovery from Advanced Databases, 8(65) (1999)
5. Pachet, F., Westermann, G., Laigre, D.: Musical data mining for electronic music distribution. In: Web Delivering of Music, Proceedings, First International Conference on IEEE, pp. 101–106 (2001)
6. Chen, L., Sakaguchi, S., Frolick, M.N.: Data mining methods, applications, and tools (2000)
7. Feldman, R., Sanger, J.: The text mining handbook: advanced approaches in analyzing unstructured data. Cambridge University Press (2007)
8. Kuri-Morales, A.F.: Minimum Database Determination and Preprocessing for Machine Learning. In: Innovative Solutions and Applications of Web Services Technology, IGI Global, pp. 94–131 (2019)
9. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. In: IEEE transactions on evolutionary computation, 6(2), pp. 182–197 (2002)
10. Kuri-Morales, A., Cartas-Ayala, A.: Polynomial multivariate approximation with genetic algorithms. In: Canadian Conference on Artificial Intelligence, Springer, Cham, pp. 307–312 (2014)

11. Cheney, E.W.: Introduction to approximation theory (1966)
12. Bezdek, J.C., Ehrlich, R., Full, W.: FCM: The fuzzy C-means clustering algorithm. Computers & Geosciences, 10(2), pp. 191–203 (1984)
13. Kohonen, T.: Self-organizing maps. Springer Science & Business Media, 30 (2001)
14. Westfall, P.H., Young, S.S.: Resampling-based multiple testing: Examples and methods for p-value adjustment. John Wiley & Sons, 279 (1993)