On choosing a suitable score function for the Bayesian Ontology Alignment tool

Vladimir Menkov

December 29, 2010

Abstract

This paper describes techniques for computing confusion matrix coefficients in the Bayesian Ontology Alignment tool (BOA). A particular attention is given to formulas that have an easy-to-understand meaning in the case of all cells of the data sources containing values from some small set, but, at the same time, can be expressed in terms of probability estimates given by a "black box" PLRM model, and thus generalized to the case of an arbitrary probability-generating model.

1 The Ontology Alignment Task

Consider the following problem. We are a given a table with N_1 rows and M_1 columns (C_1, C_2, ..., C_{M_1}), representing a sampling of data from "Data Source" DS1, i.e. something like a SQL database table. Each cell of the table contains an object from some set $\mathcal{V}$, whose exact nature we can abstract from. Each row of the table represent a structured data "record" of some kind, for example a news article, and each cell of the table correspond to a particular data element of the record's data item - e.g. the text strings containing the title, the main text, the name of the first author, the name of the second author (if any), the date, the place, etc. of the article.

There is also another table, with N_1 rows and M_1 columns ($C'_1, C'_2, \ldots, C'_{M_2}$), which represents a sampling of data from another "Data Source" DS2. The cell values in this table also belong to the set $\mathcal{V}$.

We use the notation z_{ij} and z'_{ij} for the values in the cell in row i, column j, of DS1 or DS2 respectively.

While the second table is quite different from the first, it is thought that the data in the two table correspond to the real-world objects of the same, or related types; it is also thought that the the records' data are divided in somewhat similar way into columns for the representation in the two table, even though the names of the columns in the two tables are different. Our task consists in "aligning the ontologies", i.e. figuring out which columns of the second table correspond to which columns of the first table. The results should be expressed in terms of a **"confusion matrix"**, which will contain a number for each pair (C_i, C'_j). Rows of the confusion matrix will correspond to columns of DS2, and columns of the confusion matrix, to columns of DS1.

We may consider the set of possible cell values $\mathcal{V}$ to be a subset of some finite-dimensional linear space $\mathcal{U}$, although in a special case we are going to consider that fact will be only of a limited importance.

2 Algorithm overview

The family of ontology alignment algorithms we'll be considering will be based on underlying algorithms that can in some way classifiy elements of the set $\mathcal{V}$ (i.e. set contents) with respect to their propensity to be found in various columns of DS1. Based on the numbers for individual cell contents of DS2, we then try to compute plausible "confusion matrix" coefficients linking the column of DS2 with those of DS1.

The overall algorithmic framework can be described as follows:

1 Consider the set of M_1 fields of DS1 as a single discrimination (set of labels) with M_1 classes (one per field)

2 Create $M_1 \cdot N_1$ training examples, each example being the content of one field of one record from DS1, and carrying the class label equal to the name of the field in question. (When the data are represented with records as rows and fields as columns, each "example" introduced at this step will correspond to the content of one cell of this table).

3 Tokenize etc. each "example" somehow, converting it into a vector in some linear space (a feature vector)

4 Use some kind of Bayesian regression learning algorithm, such as one of those implemented by BOXER toolkit learner, on that set of $M_1 \cdot N_1$ "examples", to come up with a classifier model that probabilistically assigns each "example" to a "class" (i.e., a field). While predictions for individual examples may be of poor quality (e.g., when several columns are all filled with "Yes" and "No" values), this can, for example, capture the fact that the "Yes"/"No" ratio is higher in some columns than in others.

5 Create $M_2 \cdot N_2$ test examples, from the "cells" of DS2, in a similar way to Step 2. Convert each example to a feature vector, as in Step 3.

6 Apply the classifier model obtained in Step 4 to these $M_2 \cdot N_2$ test examples (cells of DS2). For each one, an array of M_1 probabilities (summing to 1.0) will be thus computed, describing the likelihood of this particular cell belonging to something similar to each column of the B.

7 For each column C'_j of the DS2 we now have N_2 arrays (one for each cell) of M_2 probabilities each. We then compute each confusion matrix value f_{ij}, describing the level of "connectedness" of C'_j with DS1's column C_i, based on the N_2 values obtained in Step 6 for the cells of C'_j. The process whereby this aggregate value f_{ji} is not specified at this time; thus there is, generally, no guarantee that $\sum_i f_{ij} = 1$, and f_{ij} can be considered as a proper probability $P(C_i|C'_j)$.

8 While the values computed in Step 7 may or may not be interpreted as probabilities, the assumption is that, for a given C'_j, a greater value corresponds to a greater degree of connectedness. We can thus pick such i that $f_{ij} > f_{kj}$ for any $k \neq i$, and say that C'_j has the closest association with C_i. In other words, we will call C_i the "best match" for C'_j.

3 Some properties of the Bayesian model

Let us assume that the learning algorithm used in Step 4 is efficient enough, and is able to construct a PLRM model very close, in terms of the log-likelyhood, to the optimal one for the circumstances. What can be said about this model? Obviusly, in general the properties of the model, and in particular the probability values $P(C_i|v)$ it will ascribe to the assignment of various elements of $\mathcal{V}$ to various columns of DS1, will depend on how the elements of $\mathcal{V}$ have been converted to feature vectors. However, under a certain - often not unjustified - assumptions, the particular feature selection and the particular linear regression algorithm won't matter much.

Assumption 1. The elements of $\mathcal{V}_\infty \subset \mathcal{V}$, the set of all values of cells of DS1, have been converted to linearly independent vectors.

The above assumption holds e.g. if each distinct cell value has a particular "shibboleth" - a word that occurs in no other cell whose entire text is different from this cell's text.

This is the case, for example, if each cell contains a single word, or is empty; our feature space consists of all words occurring in the cells, plus the special "empty" token (thanks to Paul for this idea!); and each cell's content is converted to a vector with a single co-ordinate set.

Under Assumption 1, the following holds about the optimal Bayesian model one can build:

Let $\alpha_i(v)$ be the percentage of cells in column C_i that contain the value v. (Thus, $\sum_{v\in\mathcal{V}} \alpha_i(v) = 1$). Then the Bayesian probability of assigning the value v to column C_i is

$$P(C_i|v) = \frac{\alpha_i(v)}{\sum_{j=1,\ldots,M_1} \alpha_j(v)}. \tag{1}$$

In other words, the probability of assigning a given value v to a particular column C_i is proportional to the share of the cells with v in the entire table that are located in column C_i.

4 Formulas for aggregating probabilities

(These three approaches are available in BOA with the usual combination of the `train` and `test` commands.)

Assumption 2. All cell values found in DS2 are also found somewhere in DS1.

In other words, $\mathcal{V}_\in \subset \mathcal{V}_\infty$, where $\mathcal{V}_\in$ is the set of values of cells of DS2.

The above is generally "not" the case - in when it is not the case, tokenization and conversion from $\mathcal{V}$ to the feature space *do* matter; but in order to analyze certain simpler situations we will work under this assumption. The situation we have in mind is that of very limited vocabulary, when most table columns contain essentially the same values, just in different proportions.

Similarly to the definition of $\alpha_I(v)$, let us define $\gamma_j(v)$ as the proportion of the cells in DS2's column C'_j that contain the value of v. Thus, $\sum_{v\in\mathcal{V}} \gamma_i(v) = 1$.

We will now consider how, in Step 7, individual probabilities for cells within a column can be aggregated into the confusion matrix elements for the column.

Arithmetic mean One way to compute the confusion matrix value f_{ij} would be by averaging $P(C_i|v)$ for all cells of the column C'_j, which would give us the averages

$$R(C_i, C'_j) \equiv \frac{1}{N_2} \sum_{k=1}^{N_2} P(C_i|z'_{kj}). \tag{2}$$

We can directly use these averages $R(C_i, C'_j)$ as the confusion matrix coefficients linking columns of DS1 and DS2:

$$f_{ij}^{\text{method1}} = R(C_i, C'_j) = \frac{1}{N_2} \sum_{k=1}^{N_2} P(C_i|z'_{kj}). \tag{3}$$

Under Assumptions 1 and 2, they would become

$$f_{ij}^{\text{method1}} = R(C_i, C'_j) = \sum_{\mathcal{V}} \gamma_j(v) P(C_i|v) \tag{4}$$

An advantage of this method is that $\sum_{i=1\ldots,M_1} f_{ij} = 1$, and the values can be easily interpreted as probabilities. Moreover, if Assumption 1 holds, the confusion matrix would be symmetric when the two data sources are identical (i.e, $\gamma_i(v) = \alpha_i(v)$ for all i), since in this case

$$R(C_i, C_j) = \frac{\sum_{\mathcal{V}} \alpha_i(v)\alpha_j(v)}{\sum_{j=1,\ldots,M_1} \alpha_j(v)}.$$

Geometric mean Another method is to use the geometric mean instead of the arithmetic mean, computing the confusion matrix coefficients as

$$f_{ij}^{\text{method2}} = \left(\prod_{k=1}^{N_2} P(C_i|z'_{kj}) \right)^{\frac{1}{N_2}} = \prod_{\mathcal{V}} P(C_i|v)^{\gamma_j(v)}. \tag{5}$$

Multiplying probabilities can, of course, be interpreted as adding their logarithms.

Since the geometric mean of non-negative numbers of always no greater than the arithmetic mean of non-negative numbers, we know that $f_{ij}^{\text{method2}} \leq f_{ij}^{\text{method1}}$ for all pairs of columns, and the values for a given j do not sum to 1 anymore.

Note also that the average geometric is zero when any of the participant columns is zero. Thus if even a single cell of the column C'_j contains a value that is *not* found in the column C_i of DS1, then f_{ij}^{method2} will be 0. If the cells of column C'_j mixes values in a way not seen in *any* column of DS1 — that is, for every $i \in 1, \ldots, M_1$ there is some value v found in C'_j but not found in C_i — then *every* coefficient f_{ij}^{method2} for column C'_j will be zero; that is, our method will say that C'_j is not similar at all to any column of DS1.

Cosine similarity. Both of the methods above are not particularly good when what we want to distinguish are columns that are composed of the same values and are only different by the proportions of those values. What we'd like to have is a confusion matrix whose element f_{ij} is maximized whenever the vector of $\vec{\gamma}_j$, whose components are the relative frequencies $\{\gamma_j(v)\}_{v\in\mathcal{V}}$ of various values in C'_j is the same as the vector $\vec{\alpha}_i$ of relative frequencies of various values in C_i. A natural approach here would be a weighted cosine formula,

$$f_{ij}^{\text{cosine method}} = \frac{\sum_v \alpha_i(v)\gamma_j(v)\phi(v)}{\left(\sum_{v\in\mathcal{V}} \alpha_i(v)^2\phi(v)\right)^{1/2}\left(\sum_{v\in\mathcal{V}} \gamma_j(v)^2\phi(v)\right)^{1/2}},$$

with some reasonable term-weight function $\phi(v)$.

Ideally, we would like the formula for $f_{ij}^{\text{cosine method}}$ to be computable purely on the basis of probabilities for the cells, $P(C_i|z'_{kj})$, and without explicitly using the values of α_i and γ_j. This would allow us to naturally expand the use of the formula even on the situation when Assumption 1 does not entirely hold.

Considering the formula for the Bayesian probability (1), we note that the following weight would work very well for our purpose:

$$\phi(v) = \frac{1}{\sum_{j=1,\ldots,M_1} \alpha_j(v)}.$$

This kind of weight is readily interpreted as the inverse of the overall frequency of a particular cell value in the entire table DS1. Using it gives as the following scoring formula:

$$\begin{aligned} f_{ij}^{\text{cosine method}} &= \frac{\sum_v P(C_i|v)\gamma_j(v)}{\left(\sum_{v\in\mathcal{V}} P(C_i|v)\alpha_i(v)\right)^{1/2}\left(\sum_{v\in\mathcal{V}} \gamma_j(v)^2\phi(v)\right)^{1/2}} && (6)\\ &= \frac{R(C_i, C'_j)}{R(C_i, C_i)^{1/2}\left(\sum_{v\in\mathcal{V}} \gamma_j(v)^2\phi(v)\right)^{1/2}}. && (7)\end{aligned}$$

The values $R(C_i, C'_j)$ and $R(C_i, C_i)$ are simply the arithmetic means introduced in eq. (2) above, and computable without reference to Assumption 1. Unfortunately, the last factor, $\|\vec{\gamma}_j\|_\phi = \left(\sum_{v\in\mathcal{V}} \gamma_j(v)^2\phi(v)\right)^{1/2}$ is *not* computable without reference to Assumption 1. However, this $\|\vec{\gamma}_j\|_\phi$ is a factor common to f_{ij} for all i for a given j. Thus if we simply want to rank the columns of DS1 according to their "similarity" to C'_j, we can simply compute the ratios

$$s_{ij} = \frac{R(C_i, C'_j)}{R(C_i, C_i)^{1/2}} \tag{8}$$

When the matrix of these ratios s_{ij} is reported as the confusion matrix, one can compare values within the same row of this matrix, but not between rows.

5 Unequal-size samples from different columns

(This is what's done in BOA when the option `-Dinput.empty.skip=true` is in effect, and empty cells of data sources are ignored.)

The preceding discussion was carried in the assumption that we have data from the equal number (N_1) of cells from each column of DS1. Similarly, we had N_2 cells from each column of DS2.

What if we have differently-sized samples from different columns of a data source? This situation can result from the sampling process, or this can stem from the decision to *ignore* empty cells of the data source, instead of choosing to treat them as legitimate cells containing a value (an empty string).

Now, our (sample of) of DS1's column C_i will consist of n_i cells; N_1 will be understood as $\max_i n_i$. Similarly, column C'_j of DS2 will have n'_j cells, and $N_2 = \max_j n'_j$.

How will this situation affect the formulas for the confusion matrix elements proposed above?

It appears that the formulas for the **arithmetic mean** (3) and **geometric mean** (5) of the probabilities don't need to be modified. Note that, by definition of Bayesian probabilities, if a particular (sampled) column C_l of of DS1 has exactly the same composition of values of the column C_i, but we have fewer cells in our samples from C_l than from C_i (i.e., $n_l < n_i$, than all probabilities $P(C_l|V)$ will be proportioablly smaller than $P(C_i|V)$:

$$\frac{P(C_l|V)}{P(C_i|V)} = \frac{n_l}{n_i}.$$

The arithmetic and geometric averages f_{lj} will, too, be proportionally smaller than f_{ij}, i.e. $f_{lj}/f_{ij} = n_l/n_i$.

For extending the **weighted cosine similarity** formula (6), (8) to the case of differently-sized columns, we will use a different approach: namely, we will continue defining $\alpha_i(V)$ as the ratio of the cells whose value is V among the n_i cells of column C_i. The values of $\gamma_j(V)$ will be defined similarly with respect to C'_j. We will still want to define the cosine similarity f^{sym1}_{ij} as the cosine of the angle (in the weighted-dot-product space) between the vectors $\vec{\alpha}_i$ and $\vec{\gamma}_j$; that is, if our samples of columns C_i and

C_l have exactly the same composition, even though $n_i \neq n_l$, we'll want $f_{lj}^{\text{sym1}} = f_{ij}^{\text{sym1}}$ for any C'_j.

With the above guidelines in mind, we note that, with a perfect Bayesian model,

$$P(C_i|V) = \alpha_i(V)n_i/(\sum_k \alpha_k(V)n_k),$$

and

$$R(C_i, C'_j) \equiv \frac{1}{n'_j}\sum_{l=1}^{n'_j} P(C_i|z'_l j) = n_i \sum_V \frac{\alpha_i(V)\gamma_j(V)}{\sum_k \alpha_k(V)n_k}. \tag{9}$$

We can thus define the weights

$$\phi(V) = \frac{1}{\sum_k \alpha_k(V)n_k}$$

for use in our dot product, and express dot products in terms of model probabilities,

$$(\vec{\alpha_i}, \vec{\gamma_j}) = R(C_i, C'_j)/n_i,$$

$$(\vec{\alpha_i}, \vec{\alpha_i}) = R(C_i, C_i)/n_i.$$

This gives us the following generalization for (6):

$$f_{ij}^{\text{sym1}} = \frac{1}{\sqrt{n_i}} \cdot \frac{R(C_i, C'_j)}{R(C_i, C_i)^{1/2}\|\vec{\gamma_j}\|} \tag{10}$$

where, however,

$$\|\vec{\gamma_j}\| \equiv \left(\sum_{v\in\mathcal{V}} \gamma_j(v)^2\phi(v)\right)^{1/2}$$

is not expressible in general probability terms. As we did in the equal-column-size case, we will note that the value $\|\vec{\gamma_j}\|$ is the same for all elements in the same row of the confusion matrix. We thus can generalize eq. (8) as

$$s_{ij} = \frac{N_1}{\sqrt{n_i}} \cdot \frac{R(C_i, C'_j)}{R(C_i, C_i)^{1/2}} \tag{11}$$

As with (8), when the matrix of these ratios s_{ij} is reported as the confusion matrix, one can compare values within the same row of this matrix, but not between rows.

6 A symmetric-cosine approach ("Symmetric No. 1")

(This is available in BOA with the `sym1` command.)

Here we will propose an alternative approach to that outlined in Sectons 2, 4. While somewhat "strange" in its design, it will generate a confusion matrix with two pleasant properties:

1. If the two data sources are identical, the matrix will be symmetric.
2. If the two columns C_i and C'_j are identical, the matrix element f_ij will be equal to 1.

The algorithm (outlined in the general, unequal-column-size, case) is as follows:

1. Consider the set of $M_1 + M_2$ fields of DS1 and DS2 as a single discrimination (set of labels) with $M_1 + M_2$ classes (one per field)
2. Create $N_e = \sum_{i=1}^{M_1} n_i + \sum_{i=1}^{M_2} n'_i$ training examples, each example being the content of one field of one record from DS1 or DS2, and carrying the class label based on the name of the data set combined with the name of the field in question. (When the data are represented with records as rows and fields as columns, each "example" introduced at this step will correspond to the content of one cell of this table).
3. Tokenize etc. each "example" somehow, converting it into a vector in some linear space (a feature vector)
4. Use some kind of Bayesian regression learning algorithm, such as one of those implemented by BOXER toolkit learner, on that set of N_e "examples", to come up with a classifier model that probabilistically assigns each "example" to a "class" (i.e., a field).
5. For each pair of columns from DS1+DS2, compute the confusion matrix value

$$f^{\text{sym1}} = \sqrt{\frac{R(C_i|C'_j)R(C'_j|C_i)}{R(C_i|C_i)R(C'_j|C'_j)}}, \tag{12}$$

   where the averaged probabilities $R(C_i|C'_j)$ are computed as in (9).

It can be shown that the similarity value (12) is a cosine of the angle between the vectors $\vec{\alpha}_i$ and $\vec{\gamma}_j$ in the Euclidean space where the dot product is defined with the weight

$$\phi(V) = \frac{1}{\sum_{i=1}^{M_1} \alpha_i(V) n_i + \sum_{i=1}^{M_2} \gamma_i(V) n'_i}.$$

Criticism. This approach seems sensible when Assumptions 1 and 2 hold (i.e., both data sources cells are from the same limited vocabulary). However, it may have abn unpleasant drawback in a situation when free-form texts are stored in some columns. Since the learner is allowed to train on the combined discrimination including cells from the columns of both data sources, it is rewarded for finding features that distinguish columns from the two data sources that otherwise would be viewed as more similar. E.g., if texts in the cells of column C_a of DS1 are, overall, fairly similar to those in the cells of column C'_b of DS2, but contain some "shibboleth" word not found in DS2, then a well-trained learner *will* make use of that word to construct a model that views C_a as completely distinct from C'_b.

7 A symmetric-cosine approach ("Symmetric No. 2")

(This is available in BOA with the `sym2` command.)

The approach outlined in this section does not have much of a theoretical foundation (yet, at any rate), but is also symmetric in the sense that if an ontology is matched against itself, a symmetric confusion matrix is produced.

1. Construct a PLRM model based on the cells of DS1, exactly as outlined in Sec. 2, Steps 1-4. In this section, $P(C_i|\cdot)$ and $R(C_i,\cdot)$ will refer to the probabilities of assignments to the columns of DS1, and their aggregates (defined as per eq. (9)) obtained by this DS1-based model.

2. Construct another PLRM model based on the cells of DS2, in a similar way. The notation $P'(C'_i|\cdot)$ and $R'(C'_i,\cdot)$ will refer to assignment probabilities and their aggregates obtained by this DS2-based model.

3. Apply the first model to the cells of DS2, and the second model, to the cell of DS1. Compute the confusion matrix elements as follows:

$$f_{ij}^{\mathrm{sym2}} = \sqrt{\frac{R(C_i|C'_j)R'(C'_j|C_i)}{R(C_i|C_i)R'(C'_j|C'_j)}} \tag{13}$$