

Data Enrichment Using Web APIs

Karthik Gomadam, Kunal Verma, Peter .Z.Yeh

Accenture Technology Labs San Jose, CA

Abstract

AAAI creates proceedings, working notes, and technical reports directly from electronic source furnished by the authors. To ensure that all papers in the publication have a uniform appearance, authors must adhere to the following instructions.

Problem Definition: Data Enrichment

We define data enrichment as the task of organically growing a data object using information pertinent the object across heterogeneous sources, such as enterprise databases, structured and unstructured resources on the Web, and file systems.

Definitions and Functions

Data Object Definition

A data object D to be enriched consists of two elements D_k and D_u . D_k consists of attributes whose values are known, which we define formally as $D_k = \{ \langle a, v(a), k_a, k_{v(a)} \rangle \dots \}$, where a is an attribute, $v(a)$ is the value of the attribute a , k_a is the importance of a to D_k , and $k_{v(a)}$ is the confidence in the correctness of $v(a)$. D_u consists of attributes whose values are unknown and hence the targets for enrichment. We define D_u formally as $D_u = \{ \langle a, k_a \rangle \dots \}$.

Attribute Relevance

Attribute relevance is a measure of the importance of an attribute to a data object. This relevance is used to guide the selection of appropriate data sources for enrichment (see next subsection).

Our definition of importance is based on two intuitions: 1) an attribute has high importance (and hence relevance) to a data object if it is uniquely associated with the data object and 2) an attribute has high importance if it is highly discriminative w.r.t. the instances of the data object. For example, the attribute *e-mail contact* should have high importance (and hence relevance) to the *Customer* data object because it satisfies the two intuitions above. However, the attribute

Zip should have lower relevance to the *Customer* object because it does NOT satisfy the second intuition – i.e. many customers map to the same zipcode.

The Data Enrichment Engine (i.e. DEE) captures the above intuitions formally with the following equation:

$$k_a = \frac{1}{1 + e^{-(H_T(a)+1)(P'_a - P_a)}} \quad (1)$$

where,

$$H_T(a) = - \sum_{v \in a} P_v \log P_v \quad (2)$$

P'_a is the relative frequency of classes that do not have the attribute a (in the set of all classes defined in the system) and P_a is the relative frequency of classes that have a . $H_T(a)$ is the entropy of the past T values of a , and serves as a proxy for the uniqueness of the values of a (and hence how discriminative is a). Because $H_T(a)$ is recomputed for every T values, relevance of a is also adapted over time.


Data Source Selection

One of the primary features of DEE is the ability to automatically select the next data source(s) for enrichment, given a data object.

The selection of the best source to use next must consider two important factors: 1) whether the source will be able to provide values if called, and 2) whether the source targets unknown attributes in D_u with high relevance. DEE satisfies the first factor by measuring how well known values of D match the inputs required by the source. If there is a good match, then the source is more likely to return values when it's called. DEE also considers the number of times the source has been polled before to prevent "starvation" of other sources.

DEE satisfies the second factor by measuring how many high-relevance, unknown attributes the source claims to provide. If a source claims to provide a large number of high-relevance, unknown attributes, then the system should select the source over others. The second factor serves as the selection bias.

DEE formally captures these two considerations with the following equation:


$$F_s = \frac{1}{2^{M-1}} B_s \frac{\sum_{a \in D_k \cap I_s} k_{v(a)}}{|I_s|} + \frac{\sum_{a \in D_u \cap O_s} k_a}{|D_u|} \quad (3)$$

where B_s is the base fitness score of a data source s being considered (this value is randomly set between 0.5 and 0.75 when the system is initialized), I_s is the set of input attributes to the data source, O_s is the set of output attributes from the data source, and M is the number of times the data source has been selected in the context of enriching the current data object.

The data source with the highest score F_s that also exceeds a predefined minimum threshold R is selected as the next source to use for enrichment. The selection (and hence enrichment) process will continue until either D_u is empty or there are no sources whose score F_s exceeds R .

Output Value Confidence

DEE also computes the confidence in the output value given by a data source for an unknown attribute. This confidence is determined using the following formula:

$$k_{v(a')} = \begin{cases} e^{\lambda(k_{v(a')} - 1)} & , \text{ if } k_{v(a')} \neq \emptyset \\ e^{\left(\frac{1}{|V_{a'}|} - 1\right)} W & , \text{ if } k_{v(a')} = \emptyset \end{cases} \quad (4)$$

where,

$$W = \frac{\sum_{a \in D_k \cap I_s} k_{v(a)}}{|I_s|} \quad (5)$$

and $V_{a'}$ is the set of output values returned by a data source for an unknown attribute a' . If multiple values are returned, then there is ambiguity and hence the confidence in the output should be discounted. This notion is also captured in the above equation.

The above formula also considers if an output value is corroborated by output values given by previously selected data sources. If so, then the confidence should be further increased. The λ factor is the corroboration factor (< 1.0), and defaults to 1.0.

Source Utility and Adaptation

Once a data source has been called, DEE determines the utility of the source in enriching the data object of interest. Intuitively, DEE models the utility of a data source as a “contract” – i.e. if DEE provides a data source with high confidence input values, then it’s reasonable to expect the data source to provide values for all the output attributes that it claims to target. Moreover, these values should not be generic and should have low ambiguity. If these expectations are violated, then the data source should be penalized heavily.

On the other hand, if DEE did NOT provide a data source with good inputs, then the source should be penalized minimally (if at all) if it fails to provide any useful outputs.

DEE captures this notion formally with the following equation: where O_s^+ are the output attributes from a data source for which values were returned, O_s^- are the output attributes from the same data source for which values were not returned, and $P_T(v(a))$ is the relative frequency of a value $v(a)$ for an attribute a over the past T values returned by the data source.

The utility of a data source U_s from the past n calls are then used to adjust the base fitness score of the data source. This adjustment is captured with the following equation

$$B_s = B_s + \gamma \frac{1}{n} \sum_{i=1}^n U_s(T-i) \quad (6)$$

where B_s is the base fitness score of a data source s , $U_s(T-i)$ is the utility of the data source i time steps back, and γ is the adjustment rate.

Resolving Ambiguity

Ambiguity occurs, during the enrichment process, when a data source returns multiple values for an unknown attribute. For example, given the following *Customer* data object:

(*Name* : *John.Smith*, *City* : *San.Jose*, *Occupation* : *NULL*)
(7)

a data source may return multiple values for the unknown attribute of *Occupation* (e.g. Programmer, Artist, etc).

To resolve this ambiguity, DEE will branch the original object – one branch for each returned value – and each branched object will be subsequently enriched using the same algorithm described above. Hence, a single data object may result in multiple objects at the end of the enrichment process.

DEE then determines the fitness for each resulting object using the following equation:

$$\frac{\sum_{a \in D_k \cap D_u} k_{v(a)} k_a}{|D_k \cup D_u|} \quad (8)$$

The top M objects according to this fitness are then returned to the user.