R. Ahsan, et al., J. of Design & Nature and Ecodynamics. Vol. 11, No. 3 (2016) 186–197

USING ENTITY IDENTIFICATION AND CLASSIFICATION FOR AUTOMATED INTEGRATION OF SPATIAL-TEMPORAL DATA

R. AHSAN*, R. NEAMTU* & E. RUNDENSTEINER
(*Both authors equally contributed to the work) Worcester Polytechnic Institute, USA.

ABSTRACT

Big data, crucial to answering economic, social, and political questions facing our society, tend to be diverse and distributed through various sites across the Internet. The creation of tools to integrate and analyze such data is of paramount interest. Yet the automation of these processes continues to be a great challenge. Our work rests on the observation that a great number of public data sources in domains ranging from economic to demographic, although of complex structure, often share key similarities, namely the presence of the Time and Location. Our proposed Data Integration through Object Modeling framework or *DIOM* tackles the critical problem of automating data integration from a variety of public websites by abstracting key features of multi-dimensional tables and interpreting them in the context of knowledge-centered Unified Spatial Temporal Model. Our classification-driven extractors are trained to identify and classify entities from both structured and unstructured parts of spreadsheets. The unstructured part contained in titles, headers, and footers reveals critical information, so-called *Implicit Knowledge*, crucial to the correct interpretation of data. Our experimental results on real world datasets from heterogeneous public data sources show increased accuracy by 25% compared to state-of-the-art approaches.
Keywords: big data, data extraction, data integration, information retrieval.

1 INTRODUCTION

Motivation in a data-intensive world, unlocking the power hidden in big data is crucial to making informed, evidence-based decisions. This is a lesson that many organizations had to learn the hard way in dealing with the crucial aspect of the **variety** of data.

For example, a lengthy process of collecting and analyzing historical data from different states led to success in repealing the Sales and Use Tax on computer and software services, introduced in Massachusetts in 2013. In the quest to fight this action perceived as detrimental to the business growth and economic health of the state, many organizations worked together to create an integrative data source for high-fidelity and talent competitive metrics that can be used to measure the economic competitiveness and influence policy making.

Large-scale data integration is crucial for the success of such endeavors. Data from a wide spectrum of diverse websites from the Tax Policy Center, the Census Bureau, to websites like the National Science Foundation and the Bureau of Economic Analysis had to be extracted, integrated, and warehoused. These web data sources represent valuable public knowledge ready to be leveraged for policy decision making and economic forecasting. The extraction and integration of data proved challenging and time consuming. Yet, the appetite for leveraging new data sources appears endless, so automation becomes critical to the success of building and growing rich economic indexes.

The Spreadsheet Integration Problem One obstacle in capitalizing on this wealth of knowledge is the lack of generalized automated tools for data integration. Unfortunately, while progress has been made on integration [1–3], it remains challenging and labor-intensive to integrate data of the rich variety required to answer complex societal questions. A large amount of information collected from these web sites is retrieved in the form of spreadsheets. We demonstrate that actual spreadsheets from domains like tax and economics

WIT CONFERENCES
This paper is part of the Proceedings of the International Conference on Big Data (Big Data 2016)
www.witconferences.com

ISSN: 1755-7437 (paper format), ISSN: 1755-7445 (online), http://www.witpress.com/journals
DOI: 10.2495/DNE-V11-N3-186-197

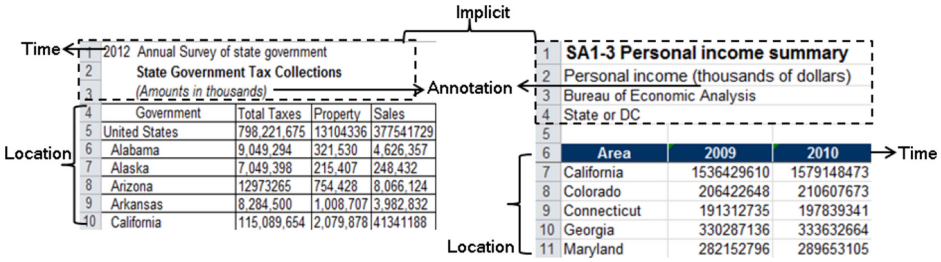

Figure 1: Conceptual similarities between spreadsheets with different structures.

while having differences in structure, organization and data representation share conceptual similarities. This is a key insight that allows us to generalize their processing in a fundamental way as we will demonstrate later.

Example: In Fig. 1 the left spreadsheet is extracted from the Tax Foundation and the right one from the Bureau of Economic Analysis. Although seemingly different, the two spreadsheets have several commonalities. In particular, they both include *Implicit knowledge* reflected in the first spreadsheet by the year "2012" and the metric "State Government Tax Collection", along with "Personal income" and the data annotation in "thousands of dollars" for the second spreadsheet. Other relevant information such as what particular state the information refers to and actual data values are located in the structured areas of both spreadsheets, henceforth referred to as *Explicit Knowledge*.

In summary, two points emerge: first, spreadsheets from diverse domains often contain multi-dimensional data based on Location, Time and Metrics. Each of these three entities can be present in any of the areas of the spreadsheet, therefore be contained in either the implicit or the explicit knowledge. This leads us to the insight of designing a *knowledge-centered three-dimensional model* to facilitate data extraction and integration for these large classes of spreadsheets.

Secondly, application datasets reveal that critical knowledge is sometimes not explicitly represented in the structured part of the spreadsheet, but rather must be inferred from unstructured regions such as titles or footnotes (*Implicit knowledge*). This implicit knowledge, if overlooked, would severely compromise the correct transformation of data and thus the resulting data integrity would be in jeopardy.

State-of-the-Art: On one end of the spectrum, tools like Talend, Knime and Oracle Data Integrator assist analysts in the data integration task by enabling them to design and define graphical mappings between source and target attributes. They would thus require users to have an in-depth knowledge of the data and to learn mapping languages [4] and operators [5]. Others [1,2] require users to specify explicit conversion rules that can be difficult and time consuming for the user to compose. At the other end of the spectrum, efforts are under way to automate various aspects of the data extraction from web spreadsheets [6,7]. Some rely heavily on physical layout features like bold and italic fonts and text indentations. Unfortunately our analysis reveals that such features are generally not present in spreadsheets in the domain we target. Lastly, computer vision techniques [3] have been applied to analyze tabular representations of spreadsheets. However, all those prior techniques tend not to focus on the knowledge hidden in unstructured parts surrounding the tabular structure. This can lead to missing key information during the integration process. To the best of our knowledge, this kind of knowledge extraction has been overlooked.

The DIOM Approach: We overcome this open challenge by our proposed Data Integration through Object Modeling (DIOM) framework that employs a rigorous spatial-temporal model to generalize the information extraction from a surprisingly large class of spreadsheets. Using the Conditional Random Field (CRF) technique [8], our DIOM entity extractor exploits knowledge from unstructured as well as structured parts of a data source. DIOM places the user at the end of the process in a reviewing role, instead of key labor-intensive steps in the middle of the integration process.

Contributions: The DIOM framework differs significantly from other systems [1,6] by exploiting both explicit and implicit categories of knowledge to ensure the correct extraction and integration of the semantics contained in spreadsheets.

1. The DIOM framework is based on a knowledge-centered three-dimensional model that serves as a foundation for abstracting key features of multi-dimensional tables. This is the first approach to leverage such spatial-temporal model to guide the automatic integration of diverse spreadsheet data.
2. Supported by the DIOM model, our entity extractor automatically identifies and classifies entities like Location and Time common to a large number of spatial-temporal spreadsheets.
3. The Data Transformation module integrates the implicit knowledge from the unstructured parts with the explicit knowledge extracted from the structured areas to compose correct information units from the spreadsheets.
4. Our comprehensive evaluation on real world data sets from four domains (economic, tax, education, and demographics) shows over 25% improvement in accuracy compared to state-of-the-art approaches.

2 RELATED WORK

We distinguish between four main approaches for extracting data from source spreadsheets into target databases.

First, the *schema-based* approach allows the users to specify the schema of spreadsheets via a layout specification language [2]. The spreadsheet data can then be converted into a database by the user explicitly specifying the source and target attribute mapping using tools such as Clio [9] or by using low-level transformation languages such as XSLT [4]. The key disadvantage in this approach is that such human-controlled mapping is specific to each spreadsheet and thus needs to be done for each spreadsheet individually. This does not scale, putting still significant manual effort in the data integration process.

Second, the *rule-based* approach requires the user to explicitly specify the transformation in the form of conversion rules [1]. The approach is flexible in that the rules could be applied to a variety of spreadsheets. However, it requires explicit conversion rules that are difficult and time consuming for the user to compose. Third, the *operator-based* approach uses database like operators on a spreadsheet interface [5]. The interface is appropriate for executing SQL queries. However, users must learn a new tool-specific proprietary language to perform transformations and extraction.

Lastly, automated approaches are the most similar to ours, [10] attempted to automatically detect errors in spreadsheets, [11] primarily focused on data normalization. More closely related to our work [6,7] uses physical layout features and hierarchical structures of the spreadsheet to extract data. Unfortunately, our analysis reveals that most spreadsheets in spatial temporal domains do not exhibit such valuable formatting characteristics. Another key difference is the technique to build the relational tuples. In [7], a relational tuple is generated by combining column headers and specific region attribute values. Our approach automatically identifies and classifies the entities present in the spreadsheet. These become the key components of the newly formed tuples, by matching our model dimensions with their physical locations in the spreadsheet.

Most importantly, none of the above approaches focuses on the extraction of knowledge from the unstructured parts of the spreadsheet, which now is the key of our proposed solution.

3 THE DIOM DESIGN

In this work, we focus on the integration of data from heterogeneous websites over a large class of application domains from economic to demographics based on the important observation that they all conceptually correspond to spatial temporal data sets. In Sec. 1, we identified a variety of such data sets from widely used public data domains. We now focus on extracting knowledge from this class of spatial-temporal spreadsheets.

3.1 DIOM data model

We now propose the spatial temporal *DIOM* model designed to handle a diversity of spreadsheets from heterogeneous data sources. The DIOM model is uniquely defined by three dimensions: Location (L), Time (T) and Metric (C).

An instance of an entity is a particular value of that entity. Generally, we denote l_i an instance of Location L, t_j an instance of Time T and c_k an instance of Metric C. Each entity can have one or multiple instances within a spreadsheet. For example, in Fig. 3 the entity Time has only one instance ("2012") while the entity Location has many instances contained in column 1, rows 5 to 10. An entity is called *singleton* if it has only one instance, otherwise it is called *composite*.

Definition 1 ***DIOM Data Model****: In the context of our spatial temporal domain, we define the DIOM data model denoted by M as a 3-dimensional model with the entities Location (L), Time (T) and Metric (C) as its dimensions:*

$$M : L \times T \times C$$

Figure 2 depicts the *DIOM* model with its dimensions based on the data presented in Fig. 3. For simplicity three instances of Location are shown, namely "United States", "Alabama" and "Alaska", three Metric instances "Total Taxes", "Property" and "Sales" and one Time instance "2012".

Definition 2 ***DIOM Data Relation:*** *A DIOM Data Relation R is a set of quadruples* (l_i, t_j, v_{ijk}) *where* $l_i \in L$, $t_j \in T$, $c_k \in C$ *and* $v_{ijk} \in V$.

$$R = \left\{ r_i = (l_i, t_j, c_k, v_{ijk}) \mid l_i \in L, t_j \in T, c_k \in C, v_{ijk} \in V \right\}$$

where r_i is an instance of the relation R.

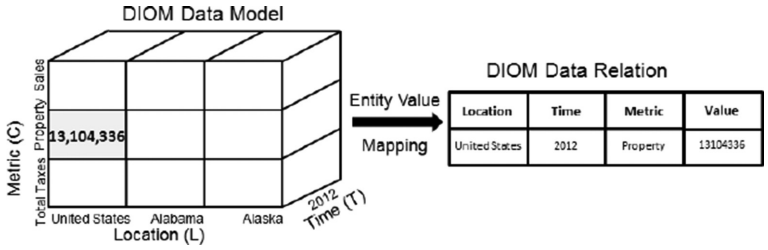

Figure 2: DIOM model.

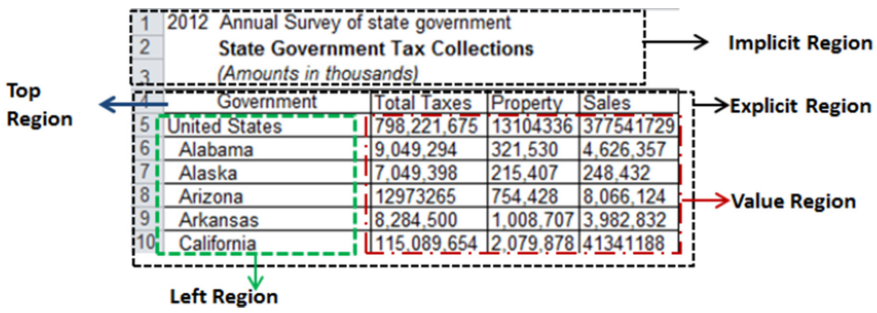

Figure 3: Labeled regions.

An example of a *DIOM Data Relation R* instance extracted from the spreadsheet in Fig. 3 is ("United States", "2012", "Total Taxes", "798,221,675").

3.2 Spreadsheet generalized templating

Let a spreadsheet be a two dimensional grid of cells $G = \{g_{ij}\}$ where i is the row index and j is the column index. Central to our methodology is the classification of the knowledge contained in the spreadsheet as *Implicit* and *Explicit*. The *Implicit Region* refers to the unstructured part. In Fig. 3, this corresponds to rows 1 to 3. The *Explicit Region* is the structured part of the spreadsheet. Inside this, the part containing the data values is designated as the *Value Region*. The part contained between the Implicit Region and the Value Region is designated as the *Top Region*, while the remaining is the *Left Region*. For example, in Fig. 3, the Explicit Region is delimited by rows 4 and 10 and columns 1 to 4. The Value Region is between rows 5 and 10 and columns 2 and 4, the Top Region is row 4, columns 1 to 4, and the Left Region is in column 1, rows 5 to 10.

For a user with domain-specific knowledge, the distinction between the two categories of knowledge is evident, yet the cost in time and resources to extract and integrate the data can be significant.

4 AUTOMATIC IDENTIFICATION AND CLASSIFICATION

The Automatic Identification and Classification module (Algorithm 1 & 2) is composed of a Region and Entity Classifier and the Meta Data Abstraction module. The purpose of the Region and Entity Classifier is to classify all the cells in the spreadsheet as region types (either implicit or explicit) and as entity types (as defined in Sec. 3). This Region and Entity classifier performs important data extraction tasks including *classification* (identifying the different types of a region e.g. *Implicit* or *Explicit*), *detection* (determining the physical position of each region) and *recognition* (locating particular entities within each region).

4.1 Region classifier

The **Region Classifier** receives a spreadsheet as input and associates labels differentiating between the regions.

Definition 3 ***Region Label:*** *The Region Classifier assigns to each row rw_j a label k, where j is the index of the row rw in the spreadsheet and*

$$k = \begin{cases} i & if\ Region = Implicit \\ e & if\ Region = Explicit \end{cases} \tag{1}$$

and the Implicit and Explicit labels reflect the associated region types previously defined (Sec. 3.2).

The cells in the Explicit Region R_e are further classified as follows: for each cell $g_{ij} \in Re$ we assign a label r_k such as:

$$r_k = \begin{cases} et & \text{if } Region = Top \\ el & \text{if } Region = Left \\ ev & \text{if } Region = Value \end{cases} \tag{2}$$

where the *Top*, *Left* and *Value* regions are defined in Sec. 3.2.

The Region Classifier employs linear-chain CRF [8] to exploit the physical layout features of the cells as well as semantic information, for example, unstructured versus structured text. The training for obtaining the semantic labels for each row of the spreadsheet is the same as in [12]. During the classification phase, each row is examined by the region classification algorithm. The rows in the unstructured part, namely the headers and footers of the file are classified as *Implicit*, while the remaining parts of the spreadsheet are categorized as *Explicit*. Any empty rows between the regions are labeled as separators. All the cells containing values are

classified as the *Value* region. The cells in the rows between the Implicit and Value region are classified as the *Top* region. The remaining cells are classified as the *Left* region. Once we have labels for each row, we can construct the correct regions.

At this moment, each cell $g_{ij} \in G$ in the spreadsheet has one or more associated labels, corresponding to the regions they belong to. We use these labels to construct "multi-label embedded vectors" (MLE-vectors) which are later used to define the region and entity boundaries (Sect. 5.3).

The classes of semantic labels are stored in an ordered list of length m, where m is the number of classes that DIOM can identify: $l = (Class_1, Class_2, ..., Class_m)$. Each class has a fixed position in the list which makes it possible to mark the association of the class in a specific position to a cell in the spreadsheet.

The list of classes identified by our system is:

i = (*Implicit, Explicit, Top, Value, Left, Time; Location, Metric, Annotation*).

Definition 4 ***Multi-label Embedded Vector (MLE-vector)***: *For each cell $g_{ij} \in G$ we construct an associated multi-label embedded vector v_{ij} of length m. The k^{th} component of the vector has the value "1" if the cell is associated with the particular class on that position in the list l, otherwise the value is "0".*

$$v_{ij}(k) = \begin{cases} 1 & if\ g_{ij}\ associated\ with\ Class_k \\ 0 & otherwise \end{cases} \tag{3}$$

In our case, for each cell in the spreadsheet, we construct a vector of length 9 (the number of our classes). For example, as shown in Fig. 3, the cell in column 1 and row 6 ("Alabama") is labeled so far as *Explicit* and *Left*.

So far, its associated *MLE-vector* is v_{16} = (0,1,0,0,1,0,0,0,0). This vector will be further updated once the entity classifier (explained in Sec. 5.2) associates other class labels with the cell.

4.2 Entity classifier

Next, we design our **Entity Classifier** to recognize the instances of entities of our spatial temporal model within each region. Exploiting the structured nature of spatial temporal spreadsheets, after the Location and Time entities are classified, the Metric can be identified by exclusion. The process starts by evaluating individual cells first on their own and later examining them in combination with neighboring cells.

Definition 5 ***Entity Label***: *The Entity Classifier assigns a label el to each grid cell $g_{ij} \in G$ where i is the row index and j is the column index of a spreadsheet*

where

$$el = \begin{cases} l & if\ Entity = Location \\ t & if\ Entity = Time \\ c & if\ Entity = Metric \\ an & if\ Entity = Annotation \end{cases} \tag{1}$$

Our Entity Classifier is a 3-class model trained to recognize: Time, Location and Annotation. The Annotation class refers to specific references like "Amounts in thousands" or "Amounts in millions". Any reference to percentage or currency of data representations (e.g., Amounts in dollars) is also considered as part of Annotation. Once the entity label has been inferred for a cell, the *MLE-vector* associated with the cell is updated to also reflect the now newly inferred labels for the entities. Referring back to the example in the Region Classifier, now the cell ("Alabama") that received the *Explicit* and *Left* labels from the region classifier, is now also assigned the *Location* label. This is reflected in the updated *MLE-vector* which is now v_{16} =(0,1,0,0,1,0,1,0,0).

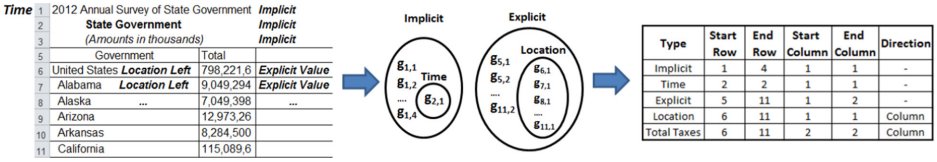

Figure 4: Example of Boundary Labels.

Similarly as with the region classifier, the CRF methodology is utilized in the design of the entity recognizer classifier. The reason for choosing this technique is two-fold: one, the technique doesn't assume that features are independent (as opposed to techniques using the Hidden Markov Model [13]) and two, future observations of the same entity type classified earlier are taken into account while labeling entity instances (as opposed to techniques using Maximum Entropy Models [4]). For example, our CRF-based classifier correctly distinguishes between an entity Location like the city "New York" and an Organization like "New York Times" that contains the same prefix as the Location entity.

Successive calls to the Entity Classifier classify each cell as Location or Time. In the Explicit region, the cells classified as Location start in row 5, column 1 and continue on the same column and subsequent rows (5 to 10).

4.3 Boundary Generator

After all the regions and entities have been identified and classified, their corresponding boundaries are set.

Definition 6 ***Region and Entity Boundary:*** *For each of the regions and entities previously specified, we define* B *as a set of boundaries*

$B = \{b_{rt}\ (sr, er, sc, ec)\}$, *where* sr =*start row,* er = *end row,* sc =*start column, ec*=*end column and* $rt \in \{rk, el\}$, *refers to the previously defined region label* rk *(Implicit or Explicit) and to the predefined entities labels el for Location, Time,*

Metric and Annotation.

The Boundary Generator uses the *MLE-vectors* to define the region and entity boundaries. To accomplish this, we perform top down hierarchical clustering of the MLE-vectors. The top-level clusters correspond to the Implicit and the Explicit regions while the next levels contain the clusters for Top, Value and Left regions. The clusters at this level are further refined as corresponding to the Time, Location, Metric and Annotation entities. Based on our observation about the contiguity of data in this class of spreadsheets, the cells in each cluster are adjacent. Thus, generating the boundaries is reduced to "decoding" the "extreme" positions of the cells in each cluster. For example, as shown in Fig. 4, the cluster of Location within the Explicit and Left clusters contains the cells $g6,1$, $g7,1$ ... $g11,1$. Thus the Location boundary attributes will be set as follows: *start-row* = 6, *end-row* = 11, *start-column* = 1, *end-column* = 1. As all the cells in the Location cluster have the same column index, the *direction* is set to "Column". For simplicity, in Fig. 4, we only show the clusters for Implicit, Explicit, Time, and Location. The direction refers to the fact that the cells labeled with Location all belong to the same column. The boundaries for Implicit, Explicit, Time and Location are shown in Fig. 4.

5 DATA TRANSFORMATION

The Data Transformation module processes the results provided by the Identification and Classification module. After the user validates the results and applies any necessary corrections, the data are extracted to generate DIOM Relations.

Algorithm 1: Find_Implicit

Input: $Spreadsheet\ S$
Output: $Region$ and $Entity\ Boundary\ \{B\}$
begin
  $B = \emptyset$
  if *cells contain unstructured text* **then**
    B=B U $implicitregionboundary$
    value=NERClassifier(cellValue);
    if *value=LOCATION* **then**
      mark implicit Location
    else if *value=TIME* **then**
      mark implicit Time
    else if *value=Annotation* **then**
      mark implicit Annotation
    else
      mark parent metric
  update cell v_{ij}
  return $\{B\}$

Algorithm 2: Mark_Explicit & Transformation

Input: $Spreadsheet\ S$, $ImplicitEndRow$
Output: $Region$ and $Entity\ Boundary\ B$
begin
  for $row \in S\ and\ row \geq ImplicitEndRow$ **do**
    if $(NERClassifier(cell.value) = Location)$ **then**
      Set Location direction
      $B = BU LocationBoundary$
    else if $(NERClassifier(cell) = Time)$ **then**
      Set Time direction B=B U $TimeBoundary$
  if $(implicitLocation)$ *or* $(implicitTime)$ **then**
    for $metricValue \in Metric$ **do**
      $T.add(Location, Time, metricValue)$
  else
    for $locationValue \in Location\ and\ timeValue \in Time$ **do**
      for $metricValue \in Metric$ **do**
        $T.add(location, time, metricValue)$
      Apply Annotation to metric

The **DIOM Relation Generator**: As shown in Algorithm 2, the DIOM Relation Generator creates the four components of an instance r_i of R as defined in Sec.3.1. The entities identified in the Implicit region are singleton entities, as explained in Sec. 3. They have to be replicated and the Relation Generator creates a number of instances equal to the product of the number of rows and columns that are in the Value region. These instances are then inserted into the DIOM Relation. For the composite entities found in the Explicit region, the instances are extracted one at the time and inserted into the DIOM Relations.

The **Value Transformer** interprets the semantic labels assigned in the Implicit region. When finding labels associated with Annotation, the Value Transformer performs the corresponding data value transformations on the tuples. For example shown in Fig. 3, the data values for all instances of metrics will be multiplied by 1,000. The user has access to specific transformation functions like multiplication that can be used as needed.

6 EXPERIMENTAL EVALUATION

Our study aims to demonstrate the improvement in accuracy and reduced human effort when using the DIOM framework compared to state-of-the-art approaches.

6.1 Experimental setup

We used two spreadsheet corpora resulted from over 2 years of collaboration with domain experts: a real world dataset **2010 Statistical Abstract of the United States (SAUS)** downloaded from the US. Census Bureau with over 1,000 files totaling 70MB and covering a variety of topics of general interest including population and income demographics and a dataset of **250 files** from websites like the National Science Foundation, the Tax Policy Center, the Bureau of Economic Analysis, the Bureau of Labor Statistics and the Tax Foundation.

Data Preprocessing: We trained and tested our model on a 50% random selection of these datasets. We randomly split the dataset into equal-sized training and testing sets. To evaluate the accuracy, we used the standard metrics of Precision, Recall and F1 [15] and repeated the split-and-test process 10 times computing the average of each metric. We asked reliable human expert users to manually examine the above spreadsheets and create ground truth region and entity labels as well as boundaries. We assumed that the users correctly labeled the

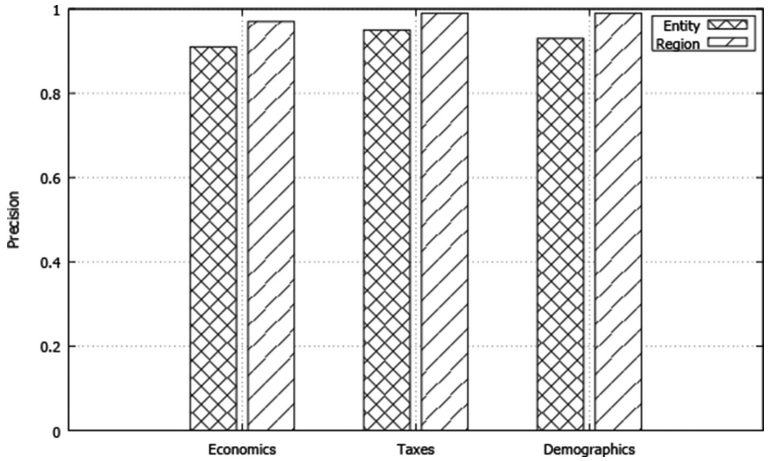

Figure 5: Region and entity classifiers accuracy.

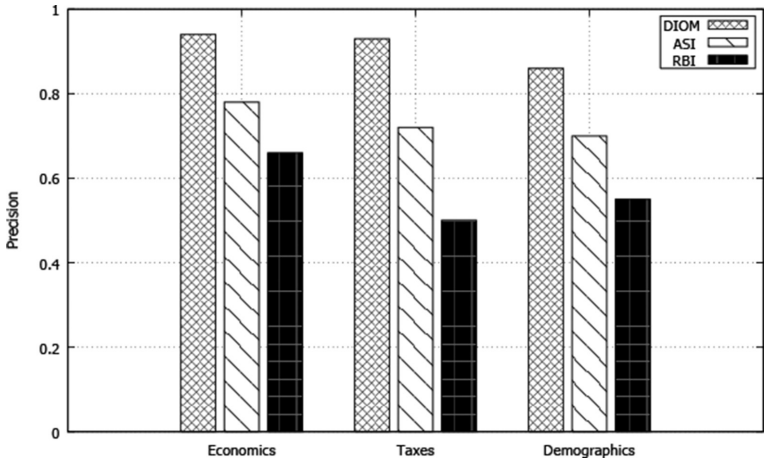

Figure 6: Data relations accuracy.

spreadsheets. In addition, we grouped the selected spreadsheets into three balanced categories based on the general topic they were related to: Economics, Taxes, and Demographics.

Our Entity Classifier extends the open source Stanford Name Entity Recognizer (NER) [16] for identifying objects in the spatial-temporal model. In particular, we expanded the functionality of the Time and Location classes for improved accuracy in recognizing the Time and Location entities as defined in our model and explained in Sec. 5.2. We added an Annotation class that also covers the existing Money and Percent classes. We trained our classifier to recognize this new class using the same training methodology as [16].

Alternate Strategies. We compared our system with the *Automatic Spreadsheet Integrator* (ASI) based on the work presented in [6,7] and the *Rule-based Integrator* (RBI) similar to [12].

The *ASI* approach uses CRF to identify the Top, Left and Value Region. Each value in the *Value Region* is combined with the annotation string from the *Top* and the *Left* regions to generate the tuples. The *RBI* parses the spreadsheet row by row after the header and treats each row in the structured part of the spreadsheet as a tuple.

We added the same code routine to both *ASI* and *RBI* to extract the *Data Relations* from the tuples generated by the each of them respectively.

6.2 Automatic extraction

During the course of our experiments, DIOM examined more than 400,000 cells and assigned the appropriate semantic labels.

The Region and Entity Classifiers. Results of the accuracy evaluation of the Region and Entity classifiers for the three previously mentioned categories are displayed in Fig. 5. It shows that the DIOM Classifiers identify the region types and entities in all three categories with an accuracy of more than 93%.

The Data Relations. The results in Table 1 and Fig. 6 show that DIOM provides better accuracy in all tested domains (25% better than *ASI* and 45% better than *RBI*).

6.3 Reduced user effort

We used a sample of randomly selected files from our datasets. We defined a metric of success for user repairs as the amount of user work reduced compared to simply fixing all the errors made by DIOM. We evaluate the human user effort by the number of corrections that a user has to perform and comparing it with correcting the errors on spreadsheets processed with DIOM versus processing spreadsheets manually by the user. We assume

Table 1: Performance of the boundary generator.

Method	Precision	Recall	F1
DIOM	0.9776	0.9776	0.9776
ASI	0.7307	0.76	0.7450
RBI	0.5769	0.60	0.5882

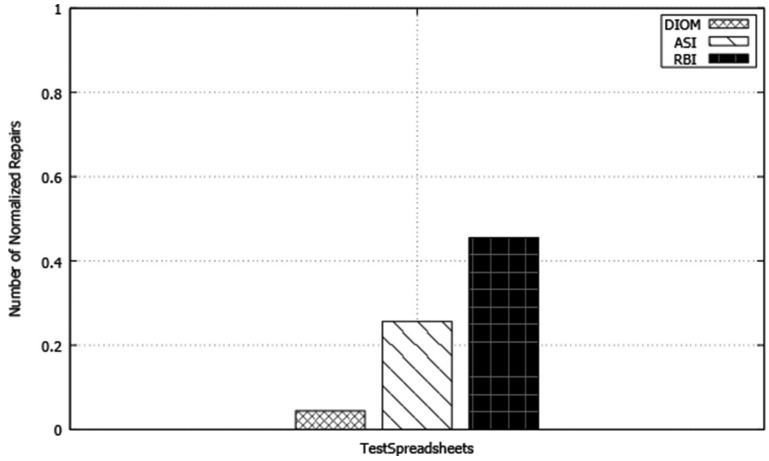

Figure 7: Normalized repairs (global).

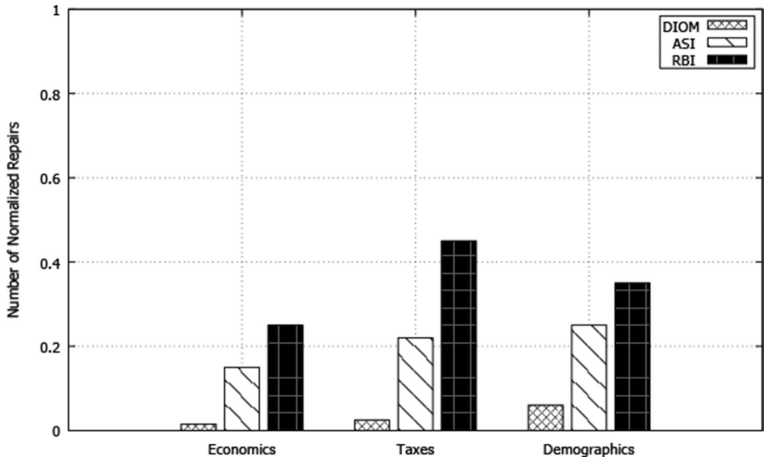

Figure 8: Normalized repairs (by category).

that the users don't make mistakes. We normalized the required repairs by dividing them by the maximum possible repairs.

Figures 7 and 8 illustrate the normalized number of repairs for all the tested spreadsheets, respectively, for each category. In both figures smaller bars correspond to better results. The savings in terms of human time and effort achieved by DIOM are major contributors in achieving automatic data integration.

7 CONCLUSION

DIOM uses the novel category of "Implicit knowledge" to automatically extract, integrate, and transform data from heterogeneous public data sources. DIOM leverages a spatial- temporal model conceptualizing on the main entity types that we found present in a large class of datasets. While we focused in this study on economic datasets, the occurrence of such datasets is rather wide spread. As demonstrated by our experimental results, the performance of our system is superior to other state-of-the-art approaches in accuracy and the user interactions are minimized.

ACKNOWLEDGMENTS

We thank Fulbright and the WPI CS department for financial support. We thank MHTC and all the contributors to the MATTERS project.

REFERENCES

[1] Hung, V., Benatallah, B. & Saint-Paul, R., Spreadsheet-based complex data transformation. In *20th ACM*, pp. 1749–1754, ACM, 2011.

[2] Lakshmanan, L.V.S., Subramanian, S.N., Goyal, N. & Krishnamurthy, R., On querying spreadsheets. In *Proceedings 14th International Conference on Data Engineering*, pp. 134–141, IEEE, 1998.

[3] Coletta, R., Castanier, E., Valduriez, P., Frisch, C., Ngo, D.H. & Bellahsene, Z., Public data integration with websmatch. In *First International Workshop on Open Data*, pp. 5–12, ACM, 2012.

[4] Roth, M., Hernandez, M.A., Coulthard, P., Yan, L., Popa, L., Ho, H.C.T. & Salter, C.C., Xml mapping technology: making connections in an xml-centric world. *IBM Systems Journal*, **45**(2), pp. 389–409, 2006.
http://dx.doi.org/10.1147/sj.452.0389

[5] Liu, B. & Jagadish, H., A spreadsheet algebra for a direct data manipulation query interface. In *Data Engineering, ICDE'09*, pp. 417–428, IEEE, 2009.
[6] Chen, Z. & Cafarella, M., Automatic web spreadsheet data extraction. In *3rd International Workshop on Semantic Search Over the Web*, p. 1. ACM, 2013.
[7] Chen, Z. & Cafarella, M., Integrating spreadsheet data via accurate and low-effort extraction. In *20th ACM SIGKDD*, pp. 1126–1135, ACM, 2014.
[8] Lafferty, J., McCallum, A. & Pereira, F.C.N., Conditional random fields: probabilistic models for segmenting and labeling sequence data, 2001.
[9] Fuxman, A., Hernandez, M.A., Ho, H., Miller, R.J., Papotti, P. & Popa, L., Nested mappings: schema mapping reloaded. In *32nd International Conference on Very Large Data Bases*, pp. 67–78, VLDB Endowment, 2006.
[10] Abraham, R. & Erwig, M., UCheck: a spreadsheet type checker for end users. *Journal of Visual Languages & Computing*, 18, pp. 71–95, 2007.
http://dx.doi.org/10.1016/j.jvlc.2006.06.001
[11] Cunha, J., Saraiva, J. & Visser, J., From spreadsheets to relational databases and back. In *2009 ACM SIGPLANworkshop on Partial Evaluation and Program Manipulation*.
[12] Pinto, D., McCullam, A., Wei, X. & Croft, W.B., Table extraction using conditional random fields. In *26th Annual International ACM SIGIR*, pp. 235–242, ACM, 2003.
[13] Malouf, R., Markov models for language-independent named entity recognition. In *6th Conference on Natural Language Learning — Volume 20*, pp. 1–4, Association for Computational Linguistics, 2002.
http://dx.doi.org/10.3115/1118853.1118872
[14] Chieu, & Ng, H.T., Named entity recognition: a maximum entropy approach using global information. In *19th International Conference on Computational Linguistics-Volume 1*, Association for Computational Linguistics, 2002.
[15] Powers, D., Evaluation: from precision, recall and f-factor to roc. *Informedness, Markedness & Correlation (Tech. Rep.)*, Adelaide, Australia, 2007.
[16] Finkel, J.R., Grenager, T. & Manning, C., Incorporating non-local information into information extraction systems by gibbs sampling. In *43rd Annual Meeting on Association for Computational Linguistics*, pp. 363–370, Association for Computational Linguistics, 2005.