Visual Segmentation for Information Extraction from Heterogeneous Visually Rich Documents

Ritesh Sarkhel
The Ohio State University
sarkhel.5@osu.edu

Arnab Nandi
The Ohio State University
arnab@cse.osu.edu

ABSTRACT

Physical and digital documents often contain visually rich information. With such information, there is no strict ordering or positioning in the document where the data values must appear. Along with textual cues, these documents often also rely on salient visual features to define distinct semantic boundaries and augment the information they disseminate. When performing information extraction (IE), traditional techniques fall short, as they use a text-only representation and do not consider the visual cues inherent to the layout of these documents. We propose VS2, a generalized approach for information extraction from heterogeneous visually rich documents. There are two major contributions of this work. First, we propose a robust segmentation algorithm that decomposes a visually rich document into a bag of visually isolated but semantically coherent areas, called logical blocks. Document type agnostic low-level visual and semantic features are used in this process. Our second contribution is a distantly supervised search-and-select method for identifying the named entities within these documents by utilizing the context boundaries defined by these logical blocks. Experimental results on three heterogeneous datasets suggest that the proposed approach significantly outperforms its text-only counterparts on all datasets. Comparing it against the state-of-the-art methods also reveal that VS2 performs comparably or better on all datasets.

CCS CONCEPTS

• **Information systems** → **Information extraction**; **Document structure**; *Content analysis and feature selection*; Entity resolution;

SIGMOD '19, June 30-July 5, 2019, Amsterdam, Netherlands

ACM ISBN 978-1-4503-5643-5/19/06…$15.00
https://doi.org/10.1145/3299869.3319867

(a) Academic event poster with highlights on the class topic, class timing and scope

(b) Real Estate flyer with highlights on the property listing, and the broker name

(c) Marketing flyer with highlights on the discount amount, product type, and target audience

Figure 1: Samples of visually rich documents using salient visual features to highlight important event information

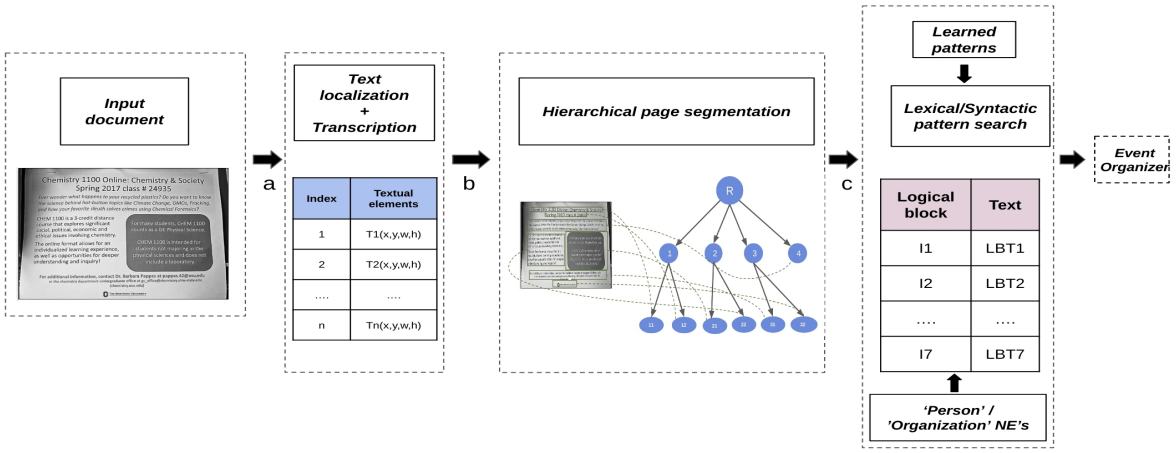
KEYWORDS

Visually Rich document; Information Extraction; Named entity

ACM Reference Format:
Ritesh Sarkhel and Arnab Nandi. 2019. Visual Segmentation for Information Extraction from Heterogeneous Visually Rich Documents. In *2019 International Conference on Management of Data (SIGMOD '19), June 30-July 5, 2019, Amsterdam, Netherlands.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3299869.3319867

1 INTRODUCTION

Information extraction from documents has been widely studied for a number of different applications in the past few years. This includes spam filtering, event extraction, social-media-text mining, text classification, and web-based document retrievals. Most of these works, however, are developed and evaluated on a text-only corpus. Consequently, contemporary literature [13, 29] on information extraction is heavily biased towards purely textual features, e.g. lexical features (e.g. stemming), contextual features (e.g., n-gram), and document features (e.g., TF-IDF). However, a significant amount of textual information that we encounter everyday is presented in the form of visually rich documents. Instead of relying solely on text-based cues, these documents use salient visual features such as text positioning, font-size distribution, typographical similarity, and color distribution to highlight or augment the semantics of various parts of the document. A few examples are shown in Fig. 1. In many cases, these documents prove to be valuable resources of useful information, not readily available in an indexed database that can be searched for a quick lookup. For example, membership



A working example of VS2 extracting the 'Event Organizer' information from an academic event poster

Figure 2: An overview of VS2; Taking a visually rich document as input, it outputs the text corresponding to a named entity from the document. Upon input, the document is cleaned and its textual elements are localized (Step a), we identify its logical blocks (Step b). This is achieved using the layout model of the document constructed using a hierarchical segmentation algorithm. Finally, a set of predefined lexical or syntactic patterns (Step c) is searched within each logical block to identify the text corresponding to the named entity to be extracted.

discount flyers from retailers, comparing online commercial real-estate enlisting by different agencies, scheduling local events from event posters; all of these scenarios require extracting structured information from visually rich documents for downstream processing. We propose a generalized method for automated information extraction from such documents in this work. To better explain our contributions, we demonstrate the limitations of a traditional information extraction (IE) method from visually rich documents in the following example.

Example 1.1: *Alice*, owner of a small event management company in Columbus, wants to survey some local events. She has collected a number of relevant event posters for this purpose and needs to extract a set of named entities $N = \{$ Event Title, Event Organizer $\}$ from these documents. For each named entity $n_i \epsilon N$, $i = 1, 2$, she wants the corresponding text t_i extracted from these documents. In scenarios such as these, a traditional text-based IE system starts with cleaning (which includes perspective warping, skew correction, and binarization) the document first. Then the document is transcribed and its text is searched for some lexical and/or syntactic patterns, predefined for each named entity to be extracted. For example, when searching for *Event Organizers* in the OCR'ed [36] transcription of an event poster, *Alice* may search for phrases that represent a 'Person' or 'Organization' in the document. If there are multiple such candidates, a word sense disambiguation strategy [30] may also be employed at this stage. Although reasonable for unstructured text, there are two major challenges in following a similar approach for visually rich event posters.

Challenge 1: Most of the natural language libraries and semantic parsers used by traditional IE methods rely on clearly defined sentence structures and context boundaries in the input text for determining various lexical and syntactic properties. Due to atypical visual properties, defining context boundaries in the transcribed text of a visually rich document may prove to be challenging (see Fig. 3). Errors introduced during optical character recognition [37, 39] of the document may adversely affect the quality of the downstream extraction task too. Now, if the document template is known beforehand, custom rules can be generated and applied on a case-by-case basis. In fact, a majority of commercially available document workflow management systems follow this scheme. These approaches, however, require a significant amount of cost and effort to maintain, making it hard to scale for diverse document types.

Challenge 2: In an information extraction workflow, entity disambiguation methods [30] are used to resolve conflicts if there are multiple matches for a named entity within the input document. Final selection is made by ranking the candidates based on some contextual information of where they appeared in the document. As traditional disambiguation strategies fail to incorporate visual features of the document, translating their success to visually rich documents (refer to Fig. 3) is also a challenging task itself.

The objective of this work is to propose a *generalized* approach for information extraction from visually rich documents. We propose VS2, a two-phase approach, to this end. Following the guidelines proposed by previous researchers [7, 35] it should have the following properties.

P1.1: Ability to intake heterogeneous documents i.e., not relying on prior knowledge about document layout or format to perform information extraction.

P1.2: Robustness i.e., flexibility to be extended for different extraction tasks.

Based on the conceptual framework by Doan, Naughton, Ramakrishnan and Baid [11], our goal is to extract a list of key-value pairs from the document. The keys originate from a predefined semantic vocabulary. VS2 retrieves the corresponding text entries from the input document. This list of key-value pairs can be loaded into a database after schema mapping. Along with traditional full-text queries, it also offers the capability to perform rich semantic queries on the document.

Upon input, VS2 starts with cleaning and localizing the textual elements of a document first. Next, it is decomposed into a number of semantically coherent visual areas, called *logical blocks* using a robust hierarchical segmentation algorithm. Identifying the logical blocks helps define the context boundaries within the document prior to any semantic parsing. Once the logical blocks have been identified, a set of lexico-syntactic patterns, defined for the named entity to be extracted, is searched within the text transcribed from each logical block. In the case of multiple matched patterns, conflict resolution is performed using a multimodal entity disambiguation strategy. The patterns are learned for each named entity, using a text-only holdout corpus. This distant supervision [1, 27] enables us to circumvent the necessity of learning extraction rules every time a document with unfamiliar layout or format is presented. In other words, it makes it easy to process large-scale heterogeneous datasets. An overview of the proposed approach is presented in Fig.2.

Technical contribution 1: Our first contribution for this purpose is a *robust encoding technique* to represent heterogeneous visually rich documents in a unified way. We propose *VS2-Segment*, a robust, hierarchical page segmentation algorithm that decomposes a document into semantically coherent visual areas, called 'logical blocks' for this purpose. Each document is represented as a bag of logical blocks. The document layout model used to enable the segmentation process will be introduced in Section 4. Further details regarding the segmentation algorithm will be provided in Section 5.1.

Technical contribution 2: Our second major contribution is a *distantly supervised search-and-select method* for identifying the named entities to be extracted within the logical blocks of a visually rich document. We propose *VS2-Select*, a distantly supervised approach that searches for a set of syntactic pattern(s) within each logical block and selects the most optimal matched pattern. In the case of multiple matches (see Fig. 3), conflict resolution is performed

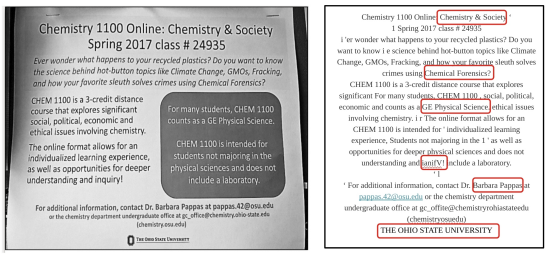


Figure 3: An illustration of the inherent challenges faced by a text-only approach for information extraction from an academic event poster sampled from dataset D2; (b) shows its transcription using Tesseract [41]; The red bounding boxes denote named entities belonging to the categories 'Person' or 'Organization' as recognized by the Stanford NER, representing potential matches for the named entity 'Event Organizer'; Most of the false positives stem from errors during transcription and ill-defined context boundaries in the transcribed text.

by a novel *optimization-based multimodal entity disambiguation* strategy. For each named entity, distances between the matched patterns and their closest *interest point*[1] in the document are minimized in a multimodal encoding space. The search-and-select method as well as the multimodal disambiguation strategy will be described in Section 5.2.

Summary of Results: We have evaluated VS2 on three heterogeneous datasets of visually rich documents. A comparative analysis against traditional text-based IE approaches (in Section 6.3 and 6.4), for separate IE tasks, reveal that the proposed method performs significantly better than these approaches on all datasets. A comparison against the state-of-the-art methods also reveals that VS2 performs better or comparable for all three tasks.

2 RELATED WORKS

Most of the early works on IE from visually rich documents take advantage of the prior knowledge of the layout of a document. One of the most popular approaches among these is wrapper induction [19, 20]. Layout specific custom masks are defined to localize and extract information from the document. Researchers like Kushmerick et al. [19] and Mironczuk et al. [28] followed this approach for IE from HTML documents. Most commercially available systems e.g., Shreddr [6], ReportMiner [22] follow a similar approach. Relying on interactive interfaces, custom rules are designed for each layout by experts and stored in a cloud-based server. For each test document, the most appropriate rule is selected manually for extracting relevant information on a case-by-case basis.

[1] An interest point is a visual area in the document that is visually and/or semantically significant

These approaches, however, are expensive to scale and hard to generalize for diverse document types. For better generalization capabilities, some recent works [9, 21, 23, 43, 46] have also proposed heuristics-based extraction grammars that leverage visual information by analyzing the layout of the document. Contrary to these methods, VS2 does not assume any prior knowledge about the layout or format of the document, making it *robust* (refer to P1.1 in Section 1) to diverse document types.

Leveraging the homogeneity of the document format, a significant fraction of the existing work exploits high-level features defined by the document markup language. In [2], HTML-specific features are used by the researchers to convert PDF documents into HTML format by assuming compliance with ISO 32000-1/32000 specifications. In a follow-up study, Gallo et al. [14] showed that this may be a strong assumption for many real-world documents that do not strictly conform with these specifications, as a slight misuse of the format operators in PDF stream may result in degraded visual descriptors in the converted HTML document. Similar limitations can be observed in [14] too. Their extractors are trained on high-level features supported by the PostScript format, making it hard to generalize for heterogeneous document formats. Unlike these methods, VS2 does not make format specific assumptions in its feature design.

Although not for IE, a computer vision based approach for segmenting web pages into coherent visual blocks was proposed by Cai et al. [4]. Each web page is recursively decomposed into smaller blocks based on a set of carefully designed rules, defined using HTML tags and the vertical and/or horizontal whitespace separators that delineate them. Compared to [4], the segmentation algorithm proposed in VS2 (detail description in Section 5.1) is more robust to diverse document types. One of the major limitations of [4] is its inability to be extended for various document formats. This is due to its reliance on HTML-specific tags to define various visual properties. Another major advantage of VS2 over [4] is its ability to segment overlapping blocks i.e., visual areas that are not separated by a rectangular (vertical/horizontal) whitespace separator. Gatterbauer et al. [15] also proposed a document-type agnostic approach for web-table construction by performing a layout analysis of rendered web pages. VS2's scope of usability is much broader than [15] as it can be applied for non-HTML documents as well as a number of non-trivial semantic tasks (refer to P1.2 in Section 1). In some of their recent works, such as Fonduer [45] and DeepDive [32], Re et al. have proposed machine-learning based solutions to this problem. A combination of visual and textual features are used to learn sequential patterns for extracting n-ary relational tuples from each document. High-level features defined by document markup languages including XML and HTML were used to train their model for this purpose. VS2 complements these methods by offering the flexibility to extend their framework for different document types, irrespective of the layout or format of the input document. Compared to the existing literature, one of the major contributions of our work lies in the fact that it is robust to diverse document layouts and formats. Contrary to most previous works, VS2 relies on a set of low-level visual and semantic features that can be extended to diverse document types (refer to *P1.1* in Section 1) to localize and extract named entities from visually rich documents. This is the first work that proposes a *generalized* approach for IE from visually rich documents and reports promising results on three heterogeneous datasets (refer to *P1.2* in Section 1) for separate information extraction tasks.

3 PROBLEM FORMULATION & DEFINITION

Suppose, we want to extract a set of named entities N= $\{n_1, n_2, ...n_p\}$ from a visually rich document D. Therefore, our objective is to return a set of textual elements t_i ($t_i \subset B$) from D, such that there is a one-to-one mapping between the text t_i and the named entity n_i, $\forall i \in [1, p]$, B denotes the set of all textual elements in D.

VS2 proposes a two-phase approach for information extraction from D. First, we represent D as a bag of visual areas $B_1, B_2, ...B_N$, such that $\{B_1, B_2, ..B_N\}$ denotes a partition[2] of B. Our core insight here is that a visually rich document is a nested object comprised of distinct visual areas that are isolated from each other but semantically coherent by themselves. Identifying these visual areas helps define the context boundaries of the document. We refer to each B_i as a *logical block* of D. Once the logical blocks are identified, VS2 searches for some predefined lexical and syntactic patterns (q_i) for each named entity n_i within the context boundaries defined by these blocks. Hence, for a set of patterns defined for the named entity ($n_i \in$ N), the task of extracting the named entity n_i can be defined as finding a mapping m, such that m: N$\rightarrow$ B. Therefore, given a document D, the task ($\mathfrak{I}$) of information extraction from D is decomposed into two sub-tasks $\mathfrak{I} = \mathfrak{I}_1 \circ \mathfrak{I}_2$.

First sub-task ($\mathfrak{I}_1$): Find a partition $P = \{B_1, B_2, ..B_N\}$ that represents D as a bag of logical blocks.

Second sub-task ($\mathfrak{I}_2$): Once P is obtained, find a mapping m: N$\rightarrow$ B , for each named entity $n_i \in$ N, that selects a set of textual elements within the boundaries defined by each $B_i \in P$, conforming to the pattern q_i.

We propose *VS2-Segment*, a hierarchical page segmentation algorithm for the first sub-task. A hierarchical layout model is constructed by the segmentation algorithm to represent the diverse visual areas within a visually rich document.

[2] $\forall i, j, B_i \subseteq B$ and $B_i \cap B_j = \phi, i \neq j$

It is discussed in greater details in the following section. The second sub-task is undertaken by *VS2-Select*, a distance supervision approach that searches for predetermined syntactic patterns within the context boundaries defined by the logical blocks and selects the most optimal matched pattern.

4 THE DOCUMENT LAYOUT MODEL

We define the layout model of a visually rich document D as a nested tuple (C, T), where C denotes the set of visual contents in D and T denotes the visual organization of D.

4.1 Visual content and their properties

An atomic element denotes the smallest unit of the visual content appearing in D. It can be classified into two major categories: *textual* and *image* element.

4.1.1 ***Textual element:*** The smallest element in a document that has textual attributes. A textual content a_t can be represented as a nested tuple, $a_t = (text\text{-}data, color, width, height)$. Here $text\text{-}data$ and $color$ represent the text appearing within a_t and the average color distribution (in LAB colorspace) of the visual area contained in a_t respectively. The attributes $height$ and $width$ denote the height and width of the smallest bounding box that encloses a_t respectively. We deem a 'word' as the textual element of a document.

4.1.2 ***Image element:*** It is an atomic element that represents an image content in the document. An image element a_i in document D is represented as a nested tuple, $a_i = (image\text{-}data, width, height)$. Here, $image\text{-}data$ denotes the image bitmap in a_i. $height$ and $width$ denote the height and width of the smallest bounding box that encloses a_i.

We have used Tesseract [41], an open-source document processing software to obtain the textual elements of a document for this work.

4.2 Organization of the visual content

The visual organization of a document is represented as a nested structure. Each visual area (v) appearing in the document is represented by the smallest bounding box (say B_v) that encloses it. Following this approach, we obtain the set of textual (A_T) and image (A_I) elements appearing in D and represent v as a string of bounding boxes enclosing the atomic elements ($A_T \cup A_I$) appearing in B_v. In this work, we represent the visual organization of a document as a tree, $T_D = \{V, E\}$, where E denotes the edges and V denotes the nodes of the tree. An edge between a parent node and its child denotes that the visual area represented by the child node is enclosed by the visual area represented by the parent node. Therefore, the *non-leaf nodes* in T_D represent the non-atomic visual areas that contain multiple smaller, semantically diverse elements within themselves. In other words, they are nested. A *leaf node*, on the other hand, corresponds to the smallest visual areas which are visually isolated but semantically coherent.

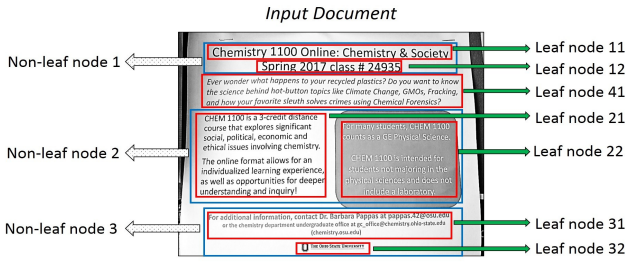


Figure 4: Each bounding box denotes a node in the layout model of the academic event poster

We represent each node v_n in T_D as a nested tuple $v_n = (B, x, y, width, height)$, where $x, y, width$, and $height$ denote the (x,y) coordinates of the left-topmost point and width and height of the smallest bounding box that encloses v_n. B denotes the set of atomic elements that appear within v_n. For a visual area v_n, B can be easily obtained by performing a reverse lookup in the list of atomic elements ($A_T \cup A_I$) appearing in D. The resulting layout tree T_D, defined this way, not only encodes the visual and semantic properties of different visual areas appearing in the document, it also captures the hierarchical relationship between them. An illustration of the layout model for an academic event poster is shown in Fig.4. The layout tree is generated by a page segmentation algorithm, employed by VS2. It will be described in more details in Section 5.1.

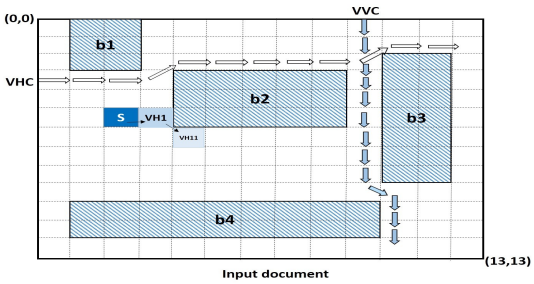
Takeaways: We propose a hierarchical layout model to represent the visual organization of visually rich documents in a unified way. The leaf-nodes of the tree-like structure represent the logical blocks of the document. Each logical block consists of a number of textual and image elements of the document, provided that they are semantically coherent. We derive the layout model using a hierarchical page segmentation algorithm called VS2-Segment.

5 OVERVIEW OF VS2

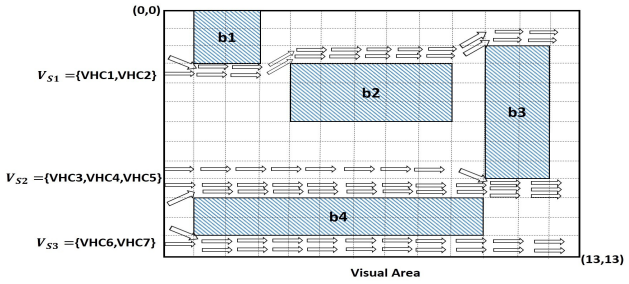
VS2 operates in two phases. In the first phase, a visually rich document is encoded as a bag of logical blocks, using *VS2-Segment*, a hierarchical page segmentation algorithm. In the next phase, *VS2-Select* searches for a set of lexical and syntactic patterns defined for each named entity, within the context boundaries of the logical blocks. We use distance supervision to learn a set of syntactic patterns for each named entity, using an isolated text-only corpus as the training dataset. The segmentation algorithm and subsequent information extraction steps are described in details in the following sections.

5.1 VS2-Segment: Segmentation of visually rich documents

The objective of VS2-Segment is to decompose a visually rich document into *logical blocks* i.e. visual areas that are semantically coherent and isolated from each other. Following



(a) $b1, b2, b3$, and $b4$ are the bounding boxes of atomic content elements in the document; S is a *whitespace position* at (2,5); $S \rightarrow VH1$ denotes a *valid 1-hop horizontal movement* from S; $S \rightarrow VH1 \rightarrow VH11$ denotes a *valid 2-hop* horizontal movement from S;The arrow-traces labeled as VHC and VVC denote a *valid horizontal* and *vertical* cut from positions (0,3) and (10,0) respectively.



(b) V_{S1}, V_{S2} and V_{S3} denote sets of consecutive valid horizontal cuts; $\{b1, b2, b3\}$, $\{b3, b4\}$, and $\{b4\}$ denote the set of neighboring bounding boxes and (0,2), (0,8), and (0,11) are the starting positions of V_{S1}, V_{S2} and V_{S3} respectively.

Figure 5: Illustrative example of the terminologies used in this work

the definitions introduced in Section 4.1, we represent each visual area by the smallest bounding box that encloses it. A bounding box $b \epsilon B$ is defined as follows, $b = (x_b, y_b, w_b, h_b)$, where x_b, y_b denote the x-y coordinate of the top-left corner and w_b, h_b denote the width and height of the bounding box. In the following section, we will define a few key terms used in the segmentation algorithm before describing the algorithm itself in Section 5.1.2.

5.1.1 *Definitions:*

- If (x, y) denote the coordinates of a position on document D such that $(x, y) \notin b, \forall\, b \epsilon B$, where B is defined above, then (x, y) is called a **whitespace position**.
- If (x, y) and $(x{+}1, y)$ are two whitespace positions in D, then a **valid horizontal movement** from (x, y) exists. If (x, y) is a whitespace position, $(x{+}1, y)$ is not a whitespace position but either of $(x{+}1,y{+}1)$ and $(x{+}1,y{-}1)$ is a whitespace position in D, then also a **valid horizontal movement** from (x, y) to that *whitespace position* exists. A *valid vertical movement* from (x, y) to a *whitespace position* between $(x, y{+}1)$, $(x{+}1, y{+}1)$ or $(x{-}1, y{+}1)$ in D, can be defined in the same way. A **valid horizontal** or **vertical movement** from (x, y) is also referred to as a **valid 1-hop movement**.
- If a horizontal movement from (x, y) to any one of the positions $(x{+}1, y{-}1)$, $(x{+}1, y)$ and $(x{+}1, y{+}1)$ is *valid* and there also is a valid horizontal movement originating from that position, then a **valid 2-hop horizontal movement** from the position (x, y) exists. Extending this definition, a **valid k-hop horizontal** or **vertical movement** from (x, y) can be easily defined.
- For a document with height H and width W, if a valid W-hop horizontal movement from $(0, y)$, $y \epsilon [0, H{-}1]$ exists, then a **horizontal cut** originating from $(0, y)$ exists. Similarly, if a valid H-hop vertical movement from $(x, 0)$, $x \epsilon [0, W{-}1]$ exists, then a **vertical cut** from $(x, 0)$ exists.

Illustrative examples of the terms introduced above are presented in Fig. 5.a and Fig. 5.b. Grid lines represent the rectangular coordinate system with the origin at left-top corner.

5.1.2 *The Segmentation Algorithm.*

Let, $T = (V, E)$ denotes the layout tree of a visually rich document D, where V denotes the set of atomic and non-atomic elements appearing in D and E denotes the set of edges representing pairwise relationships between these elements. $\forall b_1, b_2 \epsilon V$, an edge e exists between b_1 and b_2, only if b_2 is completely contained within b_1. We hypothesize that a visually rich document is a nested object comprised of smaller semantically coherent visual areas, called logical blocks. The objective of our segmentation algorithm is to identify the logical blocks of diverse visually rich documents in a generalizable way. This is achieved by recursively decomposing a document into smaller visual areas by identifying the explicit and implicit visual modifiers used to augment/highlight an area within a visually rich document. A set of empirically selected low-level visual and semantic features are used to encode each area for this purpose. The layout tree T acts as a unified data structure during the segmentation process. If a visual area (v) in D, represented by the node n_v in T, is segmented into a set of smaller areas $v_1, v_2, ...v_t$, then nodes n_i, $\forall i \epsilon [1, t]$ are added as child nodes of n_v in T. The same steps are again repeated for these newly added nodes $n_i, \forall i \epsilon [1, t]$ as more nodes representing visual elements in v_i are added as child nodes of n_i to T. At each iteration of the segmentation algorithm, the leaf nodes of T represent a set of isolated visual areas. Each node n_v in T is represented as a nested tuple (v_c, v_t), where $v_c \subset C$ denotes the set of atomic

Table 1: Visual features used for clustering

Visual Attribute	Description
centroid-position	Position of the bbox centroid
height	Height of the bounding box
color	Average color in LAB colorspace
angular distance	Angular distance of the bbox centroid from origin
sum of angular distances	The sum of angular distances between two bbox centroids

elements within the visual area v and v_t represents the complete sub-tree of T with n_v as root. After convergence, the visual areas represented by the leaf nodes of T constructed this way, represent the logical blocks of the document. Each iteration of the segmentation algorithm involves identifying the explicit and implicit visual modifiers within a visual area in the document, followed by a semantic merging step.

At every iteration, the algorithm begins by searching for explicit visual delimiters within a visual area. Each visual area v is scanned from top to bottom and left to right to identify sets of consecutive *valid horizontal* (H_s) and/or *vertical cuts* (V_s) (refer to Fig. 5.b) that may act as potential visual separators for semantically diverse visual elements appearing in v. If such separators exist, the visual area is divided into smaller areas delimited by those separators. For example, if V_{S1} and V_{S2} are visual separators in Fig. 5.b, the visual area is divided into three smaller areas v_1, v_2, and v_3, containing the bounding boxes $b1$, $\{b2, b3\}$ and b_4 respectively. Whether a set of consecutive, *valid horizontal* or *vertical cuts* should be considered as a visual separator is decided using Algorithm 1. Assuming that, (a) *distribution of the inter-area distance between textual elements is different from the distribution of intra-area separation*, and (b) *font size is uniform within a semantically coherent area*, this algorithm scans for irregularities in the distribution of correlation between the width (cardinality of the set of consecutive valid cuts) of a set of consecutive valid cuts and the maximum height of its *neighboring bounding boxes* in a topologically sorted order. A neighboring bounding box for a set of consecutive valid cuts is the bounding box which is at minimum Euclidean distance from the set of consecutive valid cuts (refer to Fig. 5.b). The correlation distribution between width and maximum neighboring bounding box heights for all consecutive valid cuts is scanned in a topological order (left to right and top to bottom) as the set closest to the first inflection point[3] of the distribution is identified to be a visual delimiter. Although the cut-based segmentation described above identifies explicit visual delimiters such as whitespace separators, it fails to recognize the implicit modifiers such as proximity, negative space, alignment, balance and symmetry that are often used to augment or highlight the semantics of areas within a visually rich document. To address this, a clustering of visual elements within v is introduced at this stage. If no visual delimiters are found at the end of the previous step, each atomic element is encoded using a set of low-level features and grouped into clusters based on pairwise similarity. The features used for this purpose are empirically selected and shown in Table 1. To initialize the clustering process, a 2×2 equal-partition grid is assumed on v and one atomic element from each cell of the grid is selected as the cluster center. We choose the atomic elements as cluster-center which are at the minimum average distance from the rest of the atomic elements in each grid cell. At each iteration of the clustering step, pairwise distances are computed for each cell and the atomic elements b_1 and b_2 are assigned to the same cluster, if $\{b_1, b_2\}$ is the closest neighbor-pair in the encoding space that is not visually separated by another atomic element. The clustering step terminates when no new element can be assigned to a different cluster.

[3]We derive the inflection points by solving for $D^2(f) = 0$, where f is the distribution of separator width vs. maximum neighboring-bbox-height

Algorithm 1 Identification of visual delimiters in D

procedure SEGMENT(S, B)
  $S = \{s_1, s_2, ...s_m\}$ ▷ `S:Consecutive valid cuts`
  $B = \{b_1, b_2, ...b_n\}$ ▷ `B:Textual elements in` D
  $width = \Phi$
  for $i = 1$ to m **do**
    $width_i = |s_i| \times \frac{argmax_k(height(neighbor{-}bbox_k(s_i)))}{argmax_j(height(b_j))}$
  Topologically sort S on (x,y) starting positions
  for $i = 2$ to m **do**
    $W = \{width_j, j \epsilon [1, i\text{-}1]\}$
    $H = \{argmax_k(\text{height}(neighbor_k(s_j))), j \epsilon [1, i\text{-}1]\}$
    $correlation_i = \rho(W, H)$
  Sort $s_i \epsilon S$ on $width_i$ in decreasing order
  for $i = 1$ to m **do**
    $C = C \cup correlation_i$
  $t = \text{inflection-point}(i, correlation_i), t \epsilon [1,\text{m-}1]$
  $VD = \{s_t, s_{t+1}....s_m\}$ ▷ `VD: Visual delimiters`
  return VD

We observed that this recursive segmentation process based on identifying visual delimiters, as described above, often leads to over-segmentation. Its effects are worse for heterogeneous datasets. To address this issue we introduce a *semantic merging* operation in our workflow. The *semantic contribution* of textual elements within a visual area is computed for this purpose. If the semantic contributions of two visual areas are similar, they are merged together. The

semantic contribution (sc_i) of a visual area, represented by node n_i in the document layout tree T is defined as follows:

$$sc_i = \Sigma_j \text{cos-similarity}(n_i, n_j) - \Sigma_k \text{cos-similarity}(n_i, n_k), \quad (1)$$

In Eq. 1, $\forall j, n_j = sibling(n_i), \forall k, n_k \neq sibling(n_i)$ and n_i, n_k denote nodes on the same level of the layout tree ($T = (V, E)$). We have used a pre-trained Word2Vec [26] embedding to compute the cosine similarities in this work. If the semantic contribution of a node(n_i) is greater than a threshold[4], it is merged with its sibling node (n_p), with which it has the highest semantic similarity among all of its sibling nodes, provided that n_i and n_p are not visually separated. In other words, $\forall a \epsilon v_{p,i} \Rightarrow a \epsilon n_p$ or $a \epsilon n_i$. Following this operation, nodes n_i and n_p are replaced by the merged node $n_{p,i}$ in the updated layout tree. The insight behind defining the semantic contribution of an area using Eqn. 1 is to ensure that each node in the layout tree represents a semantically distinct area, with respect to both the local and global context of where it appears in the document. The merging step terminates when no new nodes in the layout tree can be updated.

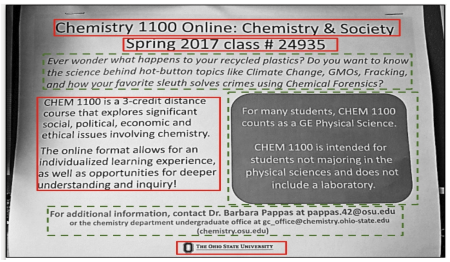


Figure 6: Each bounding box represents a logical block in the academic event poster; Visual areas enclosed by solid red bounding boxes represent interest points within the document

Takeaways: VS2-Segment maximizes the visual separation between distinct areas in the document by grouping elements that are visually similar and not separated by any visual delimiters. The explicit visual delimiters such as whitespace separators across horizontal and vertical directions of the document are identified during the beginning of every iteration. Implicit visual modifiers, on the other hand, are taken into consideration in the next phase, during a bottom-up clustering step. We observed that considering only the visual features during this process, often leads to over-segmentation. Therefore, a merging operation is undertaken that maximizes the semantic similarity between these groups by merging the visual areas that are semantically similar. At the end of this recursive process, we obtain the layout-model T. Leaf-nodes of T represent the logical blocks of the document. VS2-Segment does not assume any prior knowledge about the document template or format in any of its iterations, making it easier to be generalized for diverse document types. It is robust to rotation (upto 45°) and page artifacts that are common in many real-world scenarios. The logical blocks obtained from VS2-Segment helps define contextual boundaries, enabling effective semantic parsing within the document. An illustration of the logical blocks obtained for an academic poster is shown in Fig.6.

[4]For a layout tree of height h, the threshold parameter (θ_h) is defined as follows, $\theta_h = \theta_{min} + \frac{\theta_{max} - \theta_{min}}{10} \times h$, where $\theta_{min} = 0$ and $\theta_{max} = 1$.

Table 2: Constructing the holdout corpus

Dataset	Website	Query	Filter
D1	irs.gov	1988	1040
D2	allevents.in	NY	04/01-05/31
D2	dl.acm.org	Talks	Sorted by views
D3	fsbo.com	NY	None
D3	homesbyowner.com	NY	None

5.2 VS2-Select: Information extraction from the logical blocks

Once the layout tree (T) has been constructed, the extraction task resolves to a search-and-select operation. For every named entity to be extracted, a set of lexico-syntactic patterns is searched within the text transcribed within the contextual boundaries defined by the logical blocks. In the following section, we will describe how these patterns are learned. Furthermore, as text-based disambiguation strategies do not work well for visually rich documents, to resolve conflicts among multiple matched patterns in the previous step, we propose an optimization-based entity disambiguation strategy. Prior to these semantic operations, the transcribed text is normalized, its stopwords are removed, dependency trees are constructed, and named entities are recognized. We have used publicly available natural language processing (NLP) tools for this purpose.

5.2.1 *Learning the syntactic patterns.*

VS2-Select performs information extraction by searching for some predefined syntactic patterns within the context boundaries defined by the logical blocks. These are lexical and/or syntactic patterns learned from a holdout corpus (H). H is a readily annotated, structured, text-only corpus, constructed for the extraction task by scraping relevant public domain websites as a preprocessing step. Evidently, H consists of annotated text entries (T_{N_i}) for every named entity (N_i) related to that task i.e., $H = \bigcup_i (N_i, T_{N_i})$. We evaluate VS2 on three separate datasets in this paper.

Constructing the holdout corpus: Populating the holdout corpus H with text entries for a named entity N_i consists of four simple steps: (a) first, an expert identifies a public domain website(s) (using a web search engine) that maintain(s)

an indexed list of web pages where N_i's appear within a fixed-format HTML environment in diverse semantic contexts similar to the IE task, (b) second, select from the available filters to query the list such that the set of results returned is maximized; store the results to an HTML file, (c) extract the text T_{N_i} corresponding to N_i from all appearances of N_i in the fixed-format HTML file using a custom web-wrapper [19] and (d) finally, insert the tuples $(N_i, T_{N_i}), \forall i$ to H. For each N_i, tuples returned by querying the list were inserted to H until the distribution of distinct syntactic patterns defined by the tuples in H was approximately normal [40] or there were no more tuples to be inserted. Holdout corpus for the first IE task contained 20 tables, each with two columns, an identifier of the named entity to be extracted and its corresponding field descriptor. The holdout corpus for the second IE task was constructed from the first 500 results obtained from the search queries mentioned in Table 2. The corpus consisted of a single table with two columns, an identifier for the named entity and its corresponding text. The holdout corpus for our third task was constructed in a similar fashion by collecting the top 100 results for each search query mentioned in Table 2. A detailed description of these datasets and their corresponding IE tasks will be presented in Section 6.

Frequent sub-tree mining for learning the patterns: To identify the syntactic patterns relevant to a named entity N_i, its corresponding entry in the holdout corpus T_{N_i} is annotated with a number of handcrafted lexical and syntactic features using publicly available NLP tools. First, the text was chunked and dependency parse trees were obtained. Named entities in every chunk were identified. The named entities with category 'Location' were further augmented with a geocode tag [24]. The noun POS tags were annotated with their respective Hypernym [42] senses. Verbnet [38] senses were extracted for every Verb POS tags as well. Once these features were extracted, the maximal frequent subtrees across the chunks were obtained. We used *TreeMiner* [47], a popular frequent subtree mining algorithm for this purpose. The syntactic patterns (P_i) obtained this way represent the syntactic patterns for the named entity N_i. The patterns obtained this way for D2 and D3 are listed in Table 3 and 6. In case of D1, exact string match against the field descriptors in the holdout corpus was carried out.

Takeaways: A set of syntactic patterns are learned from a holdout corpus for each named entity to be extracted. This distance supervision approach circumvents the necessity of prior knowledge about the template or format of the document, a necessity in directly supervised approaches. This also helps avoid the curse of heterogeneity, making the proposed approach easier to generalize for diverse document types.

5.3 Entity disambiguation by optimization

Searching for a syntactic pattern within the transcribed text may result in multiple matches. This is a known phenomenon [16, 34] in IE workflows. In these scenarios, traditional IE approaches employ word-sense disambiguation [29] strategies to rank all the matches using contextual information of where they appear in the document. Due to atypical visual properties, the traditional text-based techniques, however, do not work well for visually rich documents. Hence, we propose an optimization-based disambiguation strategy in this work. Every matched pattern is encoded using a set of visual and semantic descriptors. Disambiguation is performed by minimizing the distance between a match and its closest interest point in a multimodal encoding space. The key insight here is to prioritize those matches, that are in close proximity of an 'interesting' visual area in the document. More details on identifying the interest points and the disambiguation strategy will be discussed in the following sections.

5.3.1 ***Determining the interest points.***
An *interest point* [44] represents a visual area in the document that is either visually prominent or semantically significant or both. We formulate this problem as an *optimal subset selection* [8] problem in this paper. Our objective is to select the most optimal subset from the set of all logical blocks (S_c) obtained from the document. For a logical block ($s \epsilon S_c$), we define 'optimality' using three visual and semantic objectives in our work. These are selected empirically from a number of commonly used visual or semantic modifiers used to augment or highlight the semantics of an area in a visually rich document. They are as follows:

(1) maximizing the *height of the bounding box* enclosing s; larger font size is typically used to highlight significant areas in a visually rich document
(2) maximizing semantic coherence i.e., *the sum of pairwise cosine similarities* between all text elements s and s', $\forall s, s' \epsilon S_c, s \neq s'$
(3) minimizing the *average word density*; sparsely worded visual blocks covering a significant area of the document highlight semantically significant areas in a visually rich document

We solve the subset selection problem by non-dominated sorting [25] of the universal set of logical blocks obtained by VS2-Segment. The subset of logical blocks that constituted the first-order pareto-front[5], is selected as the interest points of that document. Interest points of an academic poster, obtained this way, are shown by red bounding boxes in Fig. 6.

Takeaways: Interest points denote an optimal subset of logical blocks obtained from the segmentation algorithm.

[5] In multi-objective optimization paradigm, the pareto-front represents a state where the optimal value of one objective cannot be improved without worsening other objectives

We identify them by optimizing some visual and semantic properties, used to augment or highlight the semantics of a visual area in the document.

5.3.2 *Distance based optimization*

The semantics of a visually rich document is part of at least two modalities: textual and visual. Hence, to disambiguate among multiple matches found from the previous step, we encode every matched pattern using a set of visual and textual features and rank them based on their distances from the closest interest point in that encoding space. The distance measure is computed using Eq. 2. The candidate which is closest to an interest point in the document, is selected as the optimal match for that named entity. The features used for this purpose are empirically selected, similar to the features used to determine the interest points in a document. In the multimodal encoding space, the distance $F_{s,c}$ between two visual areas s and c is defined as follows:

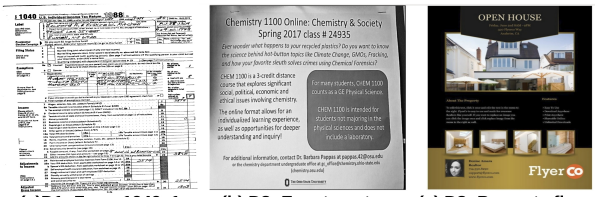
$$\mathbf{F_{s,c}} = \alpha\mathbf{\Delta D(s,c)} + \beta\mathbf{\Delta H(s,c)} + \gamma\mathbf{\Delta Sim(s,c)} + \nu\mathbf{\Delta Wd(s,c)} \quad (2)$$

where, $\alpha + \beta + \gamma + \nu$=1 and $\alpha, \beta, \gamma, \nu \epsilon [-1, 1]$. In Eqn. (3), $\Delta D(s, c)$ denotes the L1 distance between two centroids and $\Delta H(s, c)$ denotes the difference between heights of the smallest bounding-boxes enclosing the text-elements in s and c. $\Delta Sim(s, c)$ denotes the cosine similarity between text elements appearing within s and c and $\Delta Wd(s, c)$ denotes the difference between distance-normalized word-densities of the smallest bounding-boxes enclosing s and c respectively. The model parameters α, β, γ, and ν reflect the relative importance of visual saliency vs. textual verbosity in a document. For example, if the documents are not verbose but visually ornate (e.g. our second dataset), then $\alpha, \beta, \nu \geqslant \gamma$. Similarly, if the corpus is not visually rich but verbose, then $\gamma \geqslant \alpha, \beta, \nu$. For a balanced corpus (e.g. first and third datasets), it is safe to assume $\alpha \approx \beta \approx \nu \approx \gamma$. A sample from each dataset is shown in Fig. 7.

Takeaways: Once the logical blocks have been obtained for a visually rich document, VS2-Select searches for a set of lexico-syntactic patterns within these blocks, for every named entity to be extracted. These patterns are learned for each task, using distance supervision from an isolated text-only corpus. To disambiguate between multiple matches, the distance between every match and its closest interest point is minimized in a multimodal encoding space. Eq. 2 is used to compute a weighted L1 distance between two visual areas in the document for this purpose.

6 EXPERIMENTS

We evaluate VS2 for three IE tasks on three separate datasets: NIST Tax dataset (D1), Event posters dataset (D2), and Real-estate flyers dataset (D3). These datasets are heterogeneous i.e., the documents in these datasets originate from different sources, and/or belong to different types or formats. A



(a)D1: Form 1040_1 (b) D2: Event poster (c) D3: Property flyer

Figure 7: Sample documents from our experimental datasets; Figures in (a), (b) and (c) represent documents sampled from experimental datasets D1, D2 and D3

detailed description of the datasets and named entities extracted from each of them will be presented in the following section. We seek to answer three key questions in this study: (a) how does VS2 perform against a traditional text-only counterpart? (b) how does it compare against other state-of-the-art methods?, and (c) what are the individual effects of various components used in VS2 on its end-to-end extraction quality? We answer the first two questions by following a two-phase evaluation strategy. First, we evaluate the performance of *VS2-Segment* (refer to Section 6.3) in accurately locating the positions of named entities within the document. End-to-end performance is evaluated by measuring the accuracy to classify (refer to Section 6.4) the named entities accurately. In both cases, we have compared against a text-only baseline and respective state-of-the-art methods. To answer the third question, we perform an ablation study (refer to Section 6.5) to evaluate the individual effects of various components on the extraction quality for all three tasks.

6.1 Experimental datasets

We evaluate VS2 on three heterogeneous datasets (D1, D2 and D3) of visually rich documents. Datasets D2 and D3 were collected and prepared for this work.

NIST Tax dataset: Our first dataset (D1) is the NIST Tax dataset [33]. It contains 5595 images of structured tax forms, representing 20 different form faces, all of which belong to the IRS 1040 package of 1988. The IE task defined for this dataset was to extract every named entity that corresponds to a form field in the document. A complete list of the 1369 form fields defined for this dataset is available at: *https://s3.amazonaws.com/nist-srd/SD6/SD06_users_guide.pdf*.

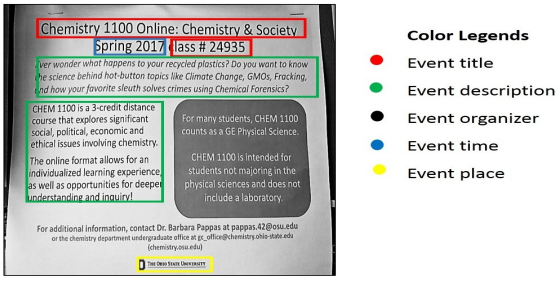


Figure 8: Ground-truth annotations of the academic event poster shown in Fig. 3

Table 3: Named entities extracted from D2

Named entity type	Description	Syntactic patterns to search
Event Title	Short description or the event	(1) Verb phrase, (2) Noun phrase with numeric (*CD*) or textual modifiers (*JJ*), and (3) SVO
Event Place	Full address of the event	Noun phrases with valid geocode tags
Event Time	Time of the event	Noun phrases with valid *TIMEX3* [5] tags
Event Organizer	Person/organization responsible for the event	(1) Verb phrase with *captain/create/reflexive_appearance* verb-senses [38], (2) Noun phrase with *Person/Organization* as named entities
Event Description	Essential details of the event (what to expect from the event if planning to attend, who will be present)	SVO **or** Verb phrase **or** Noun phrase with modifiers (*CD/JJ*)

Table 4: Named entities extracted from D3

Named entity type	Description	Syntactic patterns to search
Broker Name	Full name of the listing broker	A bigram/trigram of NE's with *Person / Organization* tags
Broker Phone	Contact number of the listing broker	A regular expression containing digits, characters and separators such as '-', '(', ')', and ' . '
Broker Email	Email address of the listing broker	An RFC-5322 compliant regular expression containing character and separators such as '@', and ' . '
Property Address	Full address information of the listing	Noun phrase with valid geocode tags
Property Size	Size-attributes summarizing the size of a listing (e.g. 4 beds, 2,465 acres)	(1) Noun phrase with numeric (*CD*) or textual modifiers (*JJ*) and (2) Noun POS tags with senses *measure / structure / estate* in the Hypernym Tree [42]
Property Description	Mentions of the property type (e.g. building,floor,land/lot) and other essential details (e.g. parking, grocery)	Noun phrases with numeric (*CD*) or textual modifiers (*JJ*)

Event posters dataset: The second dataset (D2) is a collection of event posters and flyers, advertising various local and US national events. It contains a total of 2190 event documents, collected randomly from various sources, including local magazines, bulletin boards, and event hosting websites. It contains both mobile captures of event flyers (1375 out of 2190) as well as digital flyers in PDF format (815 out of 2190). The IE task on this dataset was to extract the named entities that convey important event information. A complete list of the named entities is presented in Table 3. We have used commonly accepted lexicons [17] by NLP researchers to represent the syntactic patterns for each named entity.

Real-estate flyers dataset: Our final dataset (D3) comprises of online flyers containing commercial property listings in counties surrounding a major U.S. city. It was constructed by collecting 1200 commercial real-estate flyers from 20 different real-estate broker websites. The documents in this corpus are in HTML format. This IE task defined on D3 was to extract various attributes of the listed property. The list of the named entities is presented in Table 3. We have used commonly accepted NLP lexicons [17] to denote the syntactic patterns representing each named entity.

6.2 Evaluation metrics

Ground-truth construction: VS2 is evaluated in two phases. The performance of VS2-Segment is measured based on its *localization* capabilities i.e., locating the position of a named entity in a document. Whereas, the end-to-end performance of VS2 is evaluated based on the *accuracy* of VS2-Select to correctly identify the named entity type, post localization. We evaluate VS2 against manually annotated ground-truth data. Every document in our experimental datasets was annotated by three experts. Annotation guidelines were developed and the experts were asked to provide: (a) coordinates of the smallest bounding boxes that contained a named entity in

Table 5: Evaluation of VS2-Segment on experimental datasets

Index	Algorithm	D1		D2		D3	
		Precision (%)	*Recall* (%)	*Precision* (%)	*Recall* (%)	*Precision* (%)	*Recall* (%)
A1	*Text-only*	88.95	92.50	62.04	74.27	53.91	76.82
A2	*XY-Cut*	90.88	97.72	67.25	72.85	52.12	65.55
A3	*Voronoi-tessellation*	92.55	98.25	80.45	87.30	81.62	81.33
A4	*VIPS*	–	–	70.28	72.15	86.62	84.75
A5	*Tesseract*	77.95	86.15	74.20	80.55	79.35	83.55
A6	**VS2-Segment**	**95.50**	**98.65**	**88.26**	**87.73**	**87.67**	**84.60**

the document, and (b) a mapping between each bounding box and the named entity it contained. Annotations were performed using a specially designed web-based tool. The positional information from three experts was then averaged to derive the final coordinates. The final mapping between a bounding box and the named entity it contains was performed by majority voting among the tags assigned to that bounding box. An academic event poster from our second dataset (D2), annotated this way, is shown in Fig.8.

Two-phase evaluation: We evaluate VS2-Segment by computing intersection-over-union (IoU) between the bounding box proposals and the corresponding ground-truth annotations. Following the benchmark proposed by Everingham et at. [12] for evaluating visual object segmentation algorithms, a bounding box proposal by VS2-Segment was deemed to be accurate if its IoU score against a labeled bounding box in the ground-truth data was greater than 0.65. The labels are not considered at this stage. To measure the end-to-end extraction performance, the predicted label for all localized and semantically classified named entities are compared against their corresponding ground-truth labels. If matched, the proposal is considered to be accurate. We report precision and recall values for both phases on all experimental datasets.

6.3 Evaluation of VS2-Segment

An evaluation of VS2-Segment's performance in *accurately localizing* the named entities for all experimental datasets has been presented in Table 5. Results show that it achieves satisfactory performance for all three IE tasks. We observe relatively better performance for D1, compared to D2 and D3. This can be attributed to higher structural variability in documents belonging to datasets D2 and D3. An exhaustive error-analysis of the final results also revealed that about 80% of the errors stemmed from over-segmentation of the logical blocks due to low-quality transcription, inhibiting semantic merging at later iterations of the algorithm.

Comparison against state-of-the-art methods: We compare VS2-Segment against five contemporary page segmentation algorithms (refer to Table 5). Our first competitor (A1) is a text-based baseline method that groups words with similar word-embeddings into the same clusters. The second baseline [18] (A2) is a visual segmentation algorithm that divides a document into smaller visual areas by finding vertical and/or horizontal cuts. Our third competitor (A3) recursively segments an input document into smaller Voronoi-areas. Summary statistics such as the distribution of font size, area ratio, angular distance are taken into consideration for this purpose. VIPS by Cai et al. [4] (A4) exploits HTML-specific features to identify visual delimiters that separate visual blocks within an HTML document. All non-HTML documents were converted to HTML format. Evidently, A4 could not be applied on dataset D1. Our final baseline method (A5) is Tesseract [41], an opensource document processing software that performs hierarchical layout analysis of an input document to segment it into blocks. Results show that we were able to outperform A1, A2, A3 and A5 on all datasets. We significantly outperformed A4 on dataset D2. Most of the errors in the final segmentation result, for this baseline method, stemmed from under-segmentation of the logical blocks that were not delineated by a rectangular whitespace separator. We observed competitive results on D3.

Table 6: End-to-end evaluation of VS2 on D2

Index	Named Entity	Proposed method		
		Pr. (%)	*Rec.* (%)	$\Delta F1$(%)
N1	*Event Title*	84.88	81.09	8.98
N2	*Event Place*	76.68	86.37	3.76
N3	*Event Time*	94.67	84.70	0.49
N4	*Event Organizer*	72.56	74.41	10.50
N5	*Event Description*	76.59	86.00	1.60
	Overall	**81.08**	**82.51**	**5.07**

6.4 Evaluation of end-to-end performance

We evaluate the end-to-end performance of VS2 by measuring the accuracy of *accurately classified named entities* by VS2-Select within an input document post localization. For each dataset, we compare the performance of our method against a text-only baseline. Using Tesseract [41] to segment the input document, it searches for syntactic patterns within the text transcribed from each segmented area. Entity disambiguation is performed using Lesk [3], a state-of-the-art text-only entity disambiguation method.

Evaluation on D1: The objective of this IE task was to extract 1369 named entities corresponding to every form

Table 7: Comparison of end-to-end performance against existing methods on all datasets

Index	Algorithm	D1		D2		D3	
		Precision (%)	*Recall* (%)	*Precision* (%)	*Recall* (%)	*Precision* (%)	*Recall* (%)
A1	*ClausIE*	–	–	70.65	62.19	76.50	68.05
A2	*FSM*	85.0	90.75	77.25	79.05	84.50	82.95
A3	*ML-based*	–	–	83.92	81.0	92.65	86.40
A4	*Apostolova et al.*	92.20	96.25	85.25	85.66	87.28	90.42
A5	*ReportMiner*	96.50	100.0	51.25	62.50	67.75	80.40
A6	**VS2**	**95.25**	**98.4**	**88.05**	**85.95**	**91.80**	**90.32**

field in D1. Results show that VS2 achieved an overall average precision of **95.25%** and recall of **98.4%** for this dataset. Compared to the text-only baseline, we observed an overall improvement of **2.84%** in average F1-score.

Evaluation on D2: The objective of this task was to extract five different named entities from a corpus of visually rich event posters. The named entities and their corresponding lexico-syntactic patterns have been presented in Table 3. Table 6 presents the end-to-end evaluation of VS2 for this IE task. The final column in the table represents the average improvement in F1-score against the text-only baseline. Compared to its text-only counterpart, significant improvement in average F1-scores were observed for named entities 'Event Title' (8.98% in F1-score) and 'Event Organizer' (10.5% in F1-score). Whereas, marginal improvements were observed for 'Event Time' (0.41%). Further inspection revealed that the text-only approach performed well, if: (a) the syntactic pattern defined for a named entity contained a regular expression with partial string matching capabilities, **or** (b) there was only a single matched pattern for that entity before disambiguation. Both of these were true in this case..

Evaluation on D3: The objective of this task was to extract six different named entities from a corpus of online real-estate flyers. The named entities and their corresponding lexico-syntactic patterns are presented in Table 4. Performance of VS2 for this task is shown in Table 8. Compared to the text-only baseline, significant improvements in average F1-scores were observed for the named entities 'Broker Name' (10.18%) and 'Property Address' (4.60%). Both of these named entities were among the most visually rich entities in D3. Smaller improvements were observed for the entities 'Broker Phone' and 'Broker Email'. We observed that for most documents in D3, these named entities appeared only once in the document. Marginal improvement was observed for the entity 'Property Description' also. This was due to the low inter-annotator agreement on what constituted "essential information" of a listed property.

Compared to the text-only baseline, the average improvement in performance using VS2 was statistically significant (t-test reveals $p < 0.05$) for all datasets. Results in Table 6 and Table 8 also reveal that end-to-end performance was better on D3 compared to D2. This is due to the over-segmentation errors introduced in our workflow for the document images in D2. Low-quality transcription of some of the document images also inhibits the semantic merging step at later iterations of our segmentation algorithm leading to incorrect semantic parsing, affecting the downstream extraction task.

Table 8: End-to-end evaluation of VS2 on D3

Index	Named Entity	Proposed method		
		Pr.(%)	*Rec.* (%)	$\Delta F1$ (%)
N1	*Broker Name*	94.72	90.85	10.18
N2	*Broker Phone*	96.15	82.25	1.63
N3	*Broker Email*	97.25	95.40	2.56
N4	*Property Address*	92.68	85.50	4.60
N5	*Property Size*	85.25	93.05	3.37
N6	*Property Desc.*	84.75	94.90	0.74
	Overall	**91.80**	**90.32**	**3.84**

Comparison against existing methods: We compare the end-to-end performance of VS2 on all datasets against five contemporary information extraction methods. Results of this study are shown in Table 7.

Our first competitor is ClausIE [10], a text-only approach that constructs a set of clause-based rules for every named entity to be extracted. VS2 significantly outperforms ClausIE on both D2 and D3. It does not apply for the form field extraction task defined for dataset D1. Our second baseline is a Frequent Subtree Mining (FSM) [31, 48] approach. For every named entity to be extracted, it finds the most frequent subtrees within the dependency trees for entries against that named entity in the holdout corpus. The syntactic patterns defined by these subtrees are then searched within the transcribed text of a test document to identify the named entities. VS2 outperforms this method on all datasets. Our third competitor (Zhou et al. [49]) proposes a supervised machine-learning approach. Every non-HTML document[6] needs to be converted to HTML format for this approach. Hence it could not be applied for the first dataset D1. Due to similar reasons, we only consider those documents in D2 that are in PDF format, for a fair comparison against this method. Following this approach, an SVM based classifier

[6]PDF→HTML using http://pdftohtml.sourceforge.net/

Table 9: Evaluating individual components in VS2 by ablation study

Index	VS2-Segment		VS2-Select	ΔF1 (%)		
	Visual feature	*Semantic feature based merging*	*Entity disambiguation*	*D1*	*D2*	*D3*
A1	✓	×	✓	0.80	2.55	3.37
A2	×	✓	✓	1.07	4.22	3.84
A3	✓	✓	×	0.95	6.78	7.05
A4	✓	✓	Text-only	0.42	4.55	3.96

was trained on the dataset (60%-40% split) using some visual and textual features of the document. VS2 outperforms this method on D2 and provides comparable performance on D3. Our fourth competitor is a multimodal IE approach proposed by Apostolova et al. [2]. They proposed a combination of textual and visual features to train an SVM classifier. Results show that the proposed approach outperforms this method for all tasks. This is attributed to better semantic parsing capabilities exhibited by VS2, as it leveraged the context boundaries obtained from the prior segmentation step. Finally, we compared our method against ReportMiner[22], a commercially available, human-in-the-loop document workflow management tool. It allows its users to define custom masks for each named entity in the document. Information extraction is performed by manually selecting the most appropriate rule for a document. We randomly selected 60% of the dataset to generate the rules and evaluated our performance on the rest. Results show that this approach did not perform well for D2 and D3. Performance worsened as the variability in document layouts increased. VS2 performed competitively or better on all datasets, with lesser human effort required in its end-to-end workflow.

6.5 Ablation study

To investigate the effects of individual components in VS2 on the end-to-end extraction quality, we have performed an ablation study in this section. Each row in Table 9 measures the effect of a critical component in VS2 on overall F1-score of the downstream extraction task. The final column in Table 9 quantifies the end-to-end effect of these changes on the overall average F1-score. A1 investigates the effects of semantic merging in VS2-Segment algorithm on the overall extraction quality. Results show that although this affects the overall F1-score of all datasets, its effects are most prominent for datasets D2 and D3. This is attributed to over-segmentation of the documents, adversely affecting the localization of named entities within a document. Scenario A2 investigates the effects of incorporating visual features for segmenting a visually rich document. Similar to A1, this leads to imprecise localization of the logical blocks, contributing to poor overall F1-scores for all of our datasets. Improved results by incorporating visual features also establish our design choice of a two-phase IE method for visually rich documents. Better overall F1-scores are achieved by leveraging the contextual boundaries obtained from the prior segmentation of the document. The most useful insight gathered from this ablation study, however, stems from scenarios A3 and A4. A3 measures the effects of the proposed entity disambiguation strategy on the end-to-end extraction quality. Significant effects of this simulation are observed for all datasets. We have also compared our disambiguation strategy against Lesk [3], a popular text-based entity disambiguation method. Experimental results revealed significant improvements over the text-based disambiguation method for datasets D2 and D3.

7 CONCLUSION

We have proposed VS2, a generalized approach for information extraction from visually rich documents in this work. Using a set of empirically selected low-level features to encode each visual area, a hierarchical segmentation algorithm is proposed to divide each document into logical blocks. Named entities are extracted by following a distantly supervised search-and-select method within the contextual boundaries defined by these logical blocks. VS2 is evaluated on three heterogeneous datasets for separate IE tasks. Results suggest that careful consideration of visual and semantic features can outperform current state-of-the-art methods in end-to-end extraction quality. To the best of our knowledge, this is the first work that proposes a *generalized approach* for IE from heterogeneous visually rich documents and reports its performance on three IE tasks. In future, we would like to extend our work to incorporate more complex documents. For example, the assumption of font size similarity within each block in our current implementation can be addressed by introducing a *generalizable feature* to identify font-type. Addressing the issue of transcription errors during both phases of our workflow would improve the robustness of our method towards processing real-world documents. Extending our feature library to include sophisticated contextual semantic features (e.g. n-gram features), learning to weight each feature based on observed data, language-agnostic multimodal embedding to encode each document, would further increase the robustness of our method. We also have plans to extend this work on multilingual and nested documents in future.

REFERENCES

[1] Gabor Angeli, Julie Tibshirani, Jean Wu, and Christopher D Manning. 2014. Combining distant and partial supervision for relation extraction. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1556–1567.

[2] Emilia Apostolova and Noriko Tomuro. 2014. Combining Visual and Textual Features for Information Extraction from Online Flyers.. In *EMNLP*. 1924–1929.

[3] Satanjeev Banerjee and Ted Pedersen. 2002. An adapted Lesk algorithm for word sense disambiguation using WordNet. In *International conference on intelligent text processing and computational linguistics*. Springer, 136–145.

[4] Deng Cai, Shipeng Yu, Ji-Rong Wen, and Wei-Ying Ma. 2003. Vips: a vision-based page segmentation algorithm. (2003).

[5] Angel X Chang and Christopher D Manning. 2012. Sutime: A library for recognizing and normalizing time expressions.. In *LREC*, Vol. 2012. 3735–3740.

[6] Kuang Chen, Akshay Kannan, Yoriyasu Yano, Joseph M Hellerstein, and Tapan S Parikh. 2012. Shreddr: pipelined paper digitization for low-resource organizations. In *Proceedings of the 2nd ACM Symposium on Computing for Development*. ACM, 3.

[7] Laura Chiticariu, Yunyao Li, Sriram Raghavan, and Frederick R Reiss. 2010. Enterprise information extraction: recent developments and open challenges. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. ACM, 1257–1258.

[8] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2009. *Introduction to algorithms*. MIT press.

[9] Valter Crescenzi, Giansalvatore Mecca, Paolo Merialdo, et al. 2001. Roadrunner: Towards automatic data extraction from large web sites. In *VLDB*, Vol. 1. 109–118.

[10] Luciano Del Corro and Rainer Gemulla. 2013. ClausIE: Clause-based Open Information Extraction. In *Proceedings of the 22Nd International Conference on World Wide Web (WWW '13)*. ACM, New York, NY, USA, 355–366. https://doi.org/10.1145/2488388.2488420

[11] AnHai Doan, Jeffrey F Naughton, Raghu Ramakrishnan, Akanksha Baid, Xiaoyong Chai, Fei Chen, Ting Chen, Eric Chu, Pedro DeRose, Byron Gao, et al. 2009. Information extraction challenges in managing unstructured data. *ACM SIGMOD Record* 37, 4 (2009), 14–20.

[12] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2010. The pascal visual object classes (voc) challenge. *International journal of computer vision* 88, 2 (2010), 303–338.

[13] Emilio Ferrara, Pasquale De Meo, Giacomo Fiumara, and Robert Baumgartner. 2014. Web data extraction, applications and techniques: A survey. *Knowledge-based systems* 70 (2014), 301–323.

[14] Ignazio Gallo, Alessandro Zamberletti, and Lucia Noce. 2015. Content extraction from marketing flyers. In *International Conference on Computer Analysis of Images and Patterns*. Springer, 325–336.

[15] Wolfgang Gatterbauer, Paul Bohunsky, Marcus Herzog, Bernhard Krüpl, and Bernhard Pollak. 2007. Towards domain-independent information extraction from web tables. In *Proceedings of the 16th international conference on World Wide Web*. ACM, 71–80.

[16] Jing Jiang. 2012. Information extraction from text. In *Mining text data*. Springer, 11–41.

[17] Dan Jurafsky. 2000. *Speech & language processing*. Pearson Education India.

[18] Mukkai Krishnamoorthy, George Nagy, Sharad Seth, and Mahesh Viswanathan. 1993. Syntactic segmentation and labeling of digitized pages from technical journals. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15, 7 (1993), 737–747.

[19] Nicholas Kushmerick. 2000. Wrapper induction: Efficiency and expressiveness. *Artificial Intelligence* 118, 1-2 (2000), 15–68.

[20] Alberto HF Laender, Berthier A Ribeiro-Neto, Altigran S Da Silva, and Juliana S Teixeira. 2002. A brief survey of web data extraction tools. *ACM Sigmod Record* 31, 2 (2002), 84–93.

[21] Wei Liu, Xiaofeng Meng, and Weiyi Meng. 2010. Vide: A vision-based approach for deep web data extraction. *IEEE Transactions on Knowledge and Data Engineering* 22, 3 (2010), 447–460.

[22] Astera LLC. 2018. ReportMiner: A Data Extraction Solution. https://www.astera.com/products/report-miner. (2018). Accessed: 2018-09-30.

[23] Tomohiro Manabe and Keishi Tajima. 2015. Extracting logical hierarchical structure of HTML documents based on headings. *Proceedings of the VLDB Endowment* 8, 12 (2015), 1606–1617.

[24] Google Maps. 2018. Google Maps Api. https://developers.google.com/maps. (2018). Accessed: 2018-09-30.

[25] R Timothy Marler and Jasbir S Arora. 2004. Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization* 26, 6 (2004), 369–395.

[26] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[27] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*. Association for Computational Linguistics, 1003–1011.

[28] Marcin Michał Mirończuk. 2018. The BigGrams: the semi-supervised information extraction system from HTML: an improvement in the wrapper induction. *Knowledge and Information Systems* 54, 3 (2018), 711–776.

[29] David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes* 30, 1 (2007), 3–26.

[30] Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)* 41, 2 (2009), 10.

[31] Dat PT Nguyen, Yutaka Matsuo, and Mitsuru Ishizuka. 2007. Relation extraction from wikipedia using subtree mining. In *Proceedings of the National Conference on Artificial Intelligence*, Vol. 22. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 1414.

[32] Feng Niu, Ce Zhang, Christopher Ré, and Jude W Shavlik. 2012. DeepDive: Web-scale Knowledge-base Construction using Statistical Learning and Inference. *VLDS* 12 (2012), 25–28.

[33] National Institute of Standards and Technology. 2018. NIST Special Database 6. https://www.nist.gov/srd/nist-special-database-6. (2018). Accessed: 2018-09-30.

[34] Benjamin Roth, Tassilo Barth, Michael Wiegand, Mittul Singh, and Dietrich Klakow. 2014. Effective slot filling based on shallow distant supervision methods. *arXiv preprint arXiv:1401.1158* (2014).

[35] Sunita Sarawagi et al. 2008. Information extraction. *Foundations and Trends® in Databases* 1, 3 (2008), 261–377.

[36] Ritesh Sarkhel, Nibaran Das, Aritra Das, Mahantapas Kundu, and Mita Nasipuri. 2017. A multi-scale deep quad tree based feature extraction method for the recognition of isolated handwritten characters of popular Indic scripts. *Pattern Recognition* 71 (2017), 78–93.

[37] Ritesh Sarkhel, Nibaran Das, Amit K Saha, and Mita Nasipuri. 2016. A multi-objective approach towards cost effective isolated handwritten Bangla character and digit recognition. *Pattern Recognition* 58 (2016), 172–189.

[38] Karin Kipper Schuler. 2005. VerbNet: A broad-coverage, comprehensive verb lexicon. (2005).

[39] Asif Shahab, Faisal Shafait, and Andreas Dengel. 2011. ICDAR 2011 robust reading competition challenge 2: Reading text in scene images. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*. IEEE, 1491–1496.

[40] Samuel Sanford Shapiro and Martin B Wilk. 1965. An analysis of variance test for normality (complete samples). *Biometrika* 52, 3/4 (1965), 591–611.

[41] Ray Smith. 2007. An overview of the Tesseract OCR engine. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, Vol. 2. IEEE, 629–633.

[42] Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In *Advances in neural information processing systems*. 1297–1304.

[43] Fei Sun, Dandan Song, and Lejian Liao. 2011. Dom based content extraction via text density. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 245–254.

[44] Tinne Tuytelaars, Krystian Mikolajczyk, et al. 2008. Local invariant feature detectors: a survey. *Foundations and trends® in computer graphics and vision* 3, 3 (2008), 177–280.

[45] Sen Wu, Luke Hsiao, Xiao Cheng, Braden Hancock, Theodoros Rekatsinas, Philip Levis, and Christopher Ré. 2018. Fonduer: Knowledge base construction from richly formatted data. In *Proceedings of the 2018 International Conference on Management of Data (SIGMOD)*. ACM, 1301–1316.

[46] Yudong Yang, Yu Chen, and HongJiang Zhang. 2003. HTML page analysis based on visual cues. In *Web Document Analysis: Challenges and Opportunities*. World Scientific, 113–131.

[47] Mohammed J Zaki. 2002. Efficiently mining frequent trees in a forest. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 71–80.

[48] Xiaowen Zhang and Bingfeng Chen. 2017. A construction scheme of web page comment information extraction system based on frequent subtree mining. In *AIP Conference Proceedings*, Vol. 1864. AIP Publishing, 020059.

[49] Ziyan Zhou and Muntasir Mashuq. 2014. Web Content Extraction Through Machine Learning. (2014).