The Analysis and Extraction of Structure from Organizational Charts

Nikhil Manali, David Doermann, Mahesh Desai, and Pengyu Yan

Department of Computer Science and Engineering,
Institute for Artificial Intelligence, University at Buffalo

Abstract. Organizational charts also known as org charts are a crucial representation of an organization's structure and hierarchical relationships between its various components and positions. Extracting information from org charts manually can be time-consuming and error-prone. This paper presents an end-to-end and automated approach for extracting information from org charts using a combination of computer vision, deep learning, and natural language processing techniques. Furthermore, to evaluate the effectiveness of the information extracted, we propose a metric to measure completeness and hierarchical accuracy. This automated approach has the potential to improve organizational restructuring and resource utilization by providing a clear and concise representation of the organizational structure. This study serves as a foundation for further research in the field of hierarchical chart analysis.

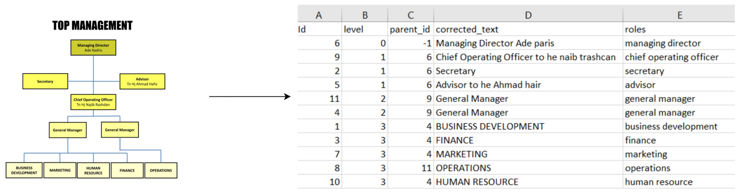
Keywords: *Org chart Analysis*, *Chart Analysis*, *Hierarchical structure*

1 Introduction

An org chart is a visual representation of the internal structure of a company that illustrates the roles, responsibilities, and relationships of individuals within the organization. It is an effective tool for understanding the hierarchical structure of a business and the relationships between different departments and positions. Org charts are commonly used during restructuring or changes in management to help employees understand how their roles fit into the overall structure of the company.

One key aspect of org charts is their ability to be analyzed for further insight. To facilitate this, it is important to represent org charts in a format that is suitable for analysis. One such format is Visio [10], which includes parent-child relationships and detailed information about each node in tabular form. However, manually converting an org chart to a tabular format requires human involvement to specify the various nodes and their hierarchy, as well as to annotate the connections between parent and child nodes. Despite the many useful applications of org charts, there is currently a lack of tools that can automatically extract and store information from these charts in a suitable format for analysis. This highlights the need for further development in this area to improve the utility of org charts in business decision-making.

Therefore, we introduce an end-to-end approach to extract and analyze an org chart and represent it into a tabular format (see Figure 1). Our main contributions to this work can be summarized as follows: (i) Discuss and formulate structural overview



Id	level	parent_id	corrected_text	roles
6	0	-1	Managing Director Ade paris	managing director
9	1	6	Chief Operating Officer to he naib trashcan	chief operating officer
2	1	6	Secretary	secretary
5	1	6	Advisor to he Ahmad hair	advisor
11	2	9	General Manager	general manager
4	2	9	General Manager	general manager
1	3	4	BUSINESS DEVELOPMENT	business development
3	3	4	FINANCE	finance
7	3	4	MARKETING	marketing
8	3	11	OPERATIONS	operations
10	3	4	HUMAN RESOURCE	human resource

Fig. 1. Org chart and its tabular representation.

of the Org chart. (ii) End-to-end approach for automatic data extraction from an org chart. (iii) Structured the stored information in a format suitable for further analysis. (iv) Evaluation metric to measure completeness and hierarchical accuracy.
We also open source our implementation and dataset for reproducibility and further research.

2 Related Work

Our work build on prior work in several domains: DETR for object detection and OTSU threshold for image binarization.

2.1 DETR

The Detection Transformer (DETR) [1] is a cutting-edge, end-to-end object detection model that streamlines the training process by treating object detection as a direct set prediction problem. It utilizes an encoder-decoder architecture based on the transformer model, incorporating its self-attention [18] mechanism to explicitly model all pairwise interactions between elements in a sequence. As noted in the original publication [1], DETR simplifies the object detection pipeline and produces a comprehensive, end-to-end solution without the need for hand-designed components that encode prior knowledge, such as spatial anchors or non-maximal suppression. The main reason for us using DETR is its end-to-end object detection and better prediction for small object/boxes.

2.2 OTSU threshold for image binarization

In our approach, image binarization is a crucial factor, particularly in the graph traversal step 4.3. The OTSU thresholding technique was utilized to determine the threshold value. This method separates the image histogram into two groups using a threshold that minimizes the weighted variance of the classes. The technique iterates through all possible threshold values, computing the spread of pixels, with the aim of finding the threshold value that results in the minimum sum of foreground and background spreads. Instead of binarizing the entire image, we used a region of interest to obtain the optimal threshold for our purposes.

3 Dataset

We collected more than 2500 org chart images from various sources, mainly online. To supplement the limited number of available samples, we augmented the original images and also generated synthetic images that mimic the structure of the org chart with varying shapes and text. As no annotated data were available and annotating an org chart is a tedious task, we developed a semi-automatic data annotation pipeline. **First**, we used computer vision techniques to detect nodes using structural kernels along with their bounding boxes. The inaccurate predictions were then filtered out and we were able to collect 1592 images which were then augmented with different augmentation techniques to increase the dataset. We also generated 500 synthetic images with bounding boxes to make our dataset more diverse and avoid overfitting. A total of 4590 samples were used to train our DETR [1] [18]model.
Given the size of the original data set used to train the DETR model *(over 65,000 images)*, we leverage transfer learning using a pre-trained DETR model with a ResNet-50 [13] base. This helped avoid overfitting and made optimal use of the training data. The training data format was structured in a manner similar to the COCO dataset [9] format used in the original DETR model training. After training, only the original org chart images were utilized for subsequent steps in the pipeline.

4 Our Approach

The analysis of an org chart requires the extraction and tabulation to be both efficient and meaningful. Our methodology involves a multistep process that efficiently extracts information from an org chart. In order to perform a meaningful analysis of an organizational chart, it is crucial to have a thorough understanding of its structure and key components. In section 4.1, we have highlighted the vital components within the organizational chart and represent its structure in a way that streamline the process of working with the chart.

4.1 Structural overview

The diagrammatic representation of an org chart serves to visually communicate the internal structure of a company, depicting the roles, responsibilities, and relationships between individuals within the entity. These diagrams use simple graphical symbols, such as lines, squares, and circles, to connect related job titles and positions within an organization. The structure of an org chart can be divided into three distinct components:

1. **Nodes or Blocks:** Each node/block represents and visually communicates a company's internal structure, the person's name, role within the organization, and possibly an image. (Figure 2)
2. **Connection Lines/Edges:** The relationships between nodes in an org chart are represented by lines connecting one node to another node(parent nodes to child nodes), which can be direct or intersect to form a junction. These connecting lines can be interpreted as directed edges from the parent node to the child node. In Figure 2 (See Image 2), a connecting line between *2* levels of the node where $level_0 > level_1$ in a hierarchy is represented as a directed edge.

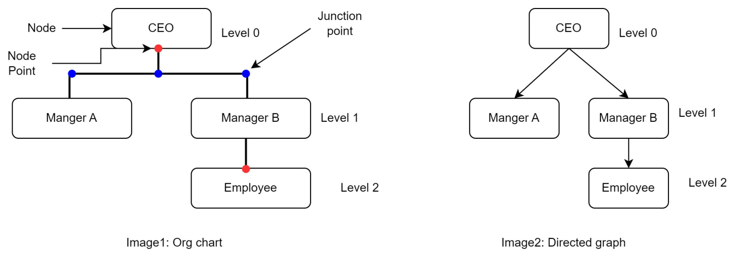


Fig. 2. org chart structural description

3. **Node and junction points:** Node points are formed at the intersections of a node and an edge, while junction points are the intersections of two edges.
   The utilization of these structural components is a crucial aspect of our approach in the subsequent stages of our work.

The process of extracting information from an org chart or any other hierarchical chart involves the following two key steps:

1. Identification of individual nodes or data blocks.
2. Analysis of the relationships between these nodes and other related nodes.

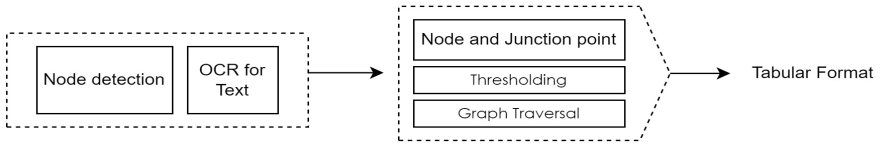


Fig. 3. org chart structural description

4.2 Localization of Node or Data blocks

Localizing or detecting nodes is crucial to determine different levels and positions in an org chart. A node, as described in Section 4.1, holds information about an individual and its position within the organization. To extract information accurately, the location of the node and its contents must be identified.

Node location: We approached this as an object detection problem, where the nodes in an org chart were considered as objects with shapes such as squares, rectangles, and ellipses. To solve this, we utilized a pre-trained object detection model from Facebook called DETR [1]. Our choice of DETR was based on its end-to-end solution for object detection using a self-attention mechanism [18], and its superior performance compared to other anchor-based models like YOLO for our particular training examples. Our training pipeline is semi-supervised and involves the automatic labeling of

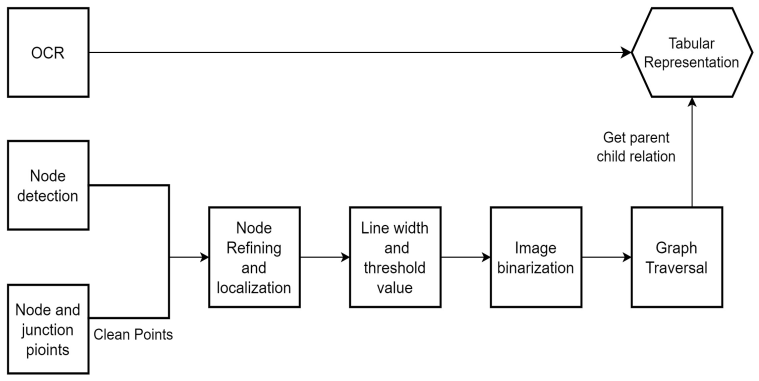

OCR
Tabular Representation
Get parent child relation
Node detection
Node Refining and localization
Line width and threshold value
Image binarization
Graph Traversal
Node and junction pioints
Clean Points

Fig. 4. org chart Analysis Pipeline

data (see Section. 3) using computer vision techniques to detect shapes. These labeled images were then used as training examples for our DETR model.
We further enhanced the localization of the bounding boxes from the DETR output by implementing corner refinement. The bounding box is defined by its corner point coordinates (x, y) and dimensions (height (H) and width (W)) as (x, y, h, w) or (x_1, y_1, x_2, y_2), where (x_1, y_1) and (x_2, y_2) represent the upper left and bottom right corners, respectively. We created a region of interest around the DETR bounding box corner and utilized the Harris corner detector to obtain precise corner points. The corner points obtained were then used to calculate new values (x_1, y_1) and (x_2, y_2), resulting in a refined bounding box.

Text detection: Once the precise locations of the nodes were obtained, we proceeded to extract the information contained within them. The information was in the form of text and, therefore, we used optical character recognition (OCR) to detect it. Specifically, we use easy-OCR [5] on the images to retrieve the text and its location and then map this information to the respective node.

4.3 Analyzing and Finding relation between nodes

The analysis and identification of the relationship between nodes is a critical aspect of understanding the hierarchical structure of an org chart. Once the precise location of the nodes and the corresponding text have been established, the next step involves determining the connections and relations between the various nodes. The accuracy of these relationships is paramount in presenting the org chart in a structured tabular format.

The org chart can be represented as a directed graph, with directed edges connecting the parent node to its child node (see Figure 2 in Image 2). To establish these relationships, we used the depth first search (DFS) for the traversal of the graph and topological ordering [2] [3] of the nodes. This allowed us to determine the parent-child relationships and their respective levels in the graph hierarchy, with the root node being at level one, its children at level two, etc.

To properly order the directed graph, it was necessary to identify the root node that holds the highest position in the organization. To accomplish this, we utilized Natural Language Processing (NLP) techniques to extract the root node from a corpus containing information about organizational positions and seniority levels. The root node represents the most senior position in the org chart.

Node and Junction points: As mentioned in section. 4.1, the Harris corner detector [7] is used to find the intersection points in a graph, which are then classified into node and junction points. We also used Shi-Tomasi [17] ***goodFeaturesToTrack()*** to refine the corner points.

Instead of $R = Scoring\ Function$ in Harris Corner

$$R = \lambda_1\lambda_2 - K(\lambda_1 + \lambda_2)^2$$

we use:

$$R = min(\lambda_1, \lambda_2)$$

1. **Node Points:** A node point is a corner point where a node intersects a connection line or edge and is classified as a Node point (denoted with a red dot in Figure 2).
2. **Junction points:** Junction points are those points intersection of two edges.

   Junction points are important because they reveal the characteristics of the connection lines or edges at their intersections. To determine which pixels in an image belong to these lines/edges, it is necessary to correctly classify corner points as either node points or junction points. Points that are not classified as node points are considered junction points. However, many unwanted junction points may be detected due to the presence of different shapes in the background of the org chart.

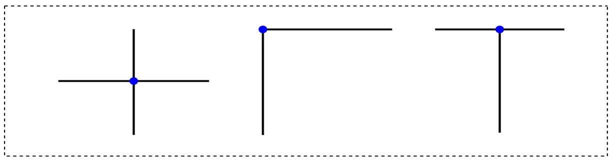


Fig. 5. Points at the cross, L and T junctions.

   To eliminate these unwanted junction points, a bounding box is formed around potential junction points and if the points do not result from the described intersection as shown in Figure 5, they are filtered as invalid junction points.

Image Thresholding: Although we can visually identify edges between our nodes, it is important to have a clear and precise classification of edges at the pixel level for graph traversal. In order to obtain this information, we utilize junction points, as they are formed at the intersection of connection lines or edges, and provide valuable information at the pixel level. We used all these junction points as a center and created a bounding box around them. Then, OTSU [11] [19] thresholding (Figure. 6) was

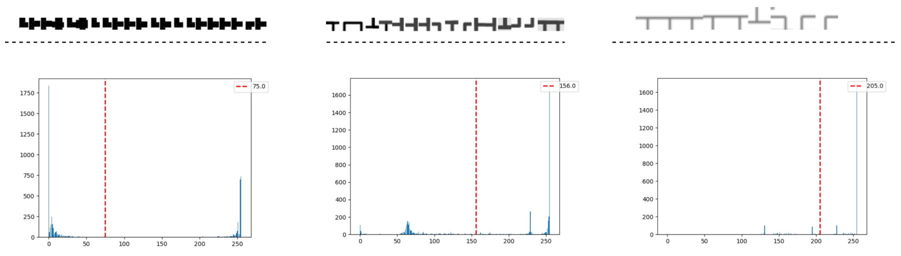


Fig. 6. A few examples of junction point regions that are used to find the threshold value using OTSU thresholding. The red lines in the figure show the division of pixel densities into two clusters, and this line represents the threshold value.

performed in these sections, generating a histogram representing the distribution of pixel intensities.

The core idea behind OTSU thresholding is to separate the image histogram into two groups with a threshold defined as a result of minimizing the weighted variance of these classes denoted by $\sigma_w^2(t)$.

The whole computation equation can be described as follows.

$$\sigma_w^2(t) = w_1(t)\sigma_1^2(t) + w_2(t)\sigma_2^2(t)$$

where $w_1(t), w_2(t)$ are the probabilities of the two classes divided by a threshold t, range from 0 to 255 inclusively.

The probability P is calculated for each value of the pixels in two separate groups C_1, C_2 using the cluster probability functions expressed as:

$$w_1(t) = \sum_{i=1}^{t} P(i), w_2(t) = \sum_{i=t+1}^{I} P(i)$$

After applying OTSU thresholding, the image was binarized [11] into foreground and background segments. The foreground was represented by pixels with a value of 0, while the background was represented by pixels with a value of 255.

Graph Traversal: At this point, we have also established a connected graph consisting of nodes and edges, and using the thresholding technique, as outlined in Section 4.3, we have identified the pixels that belong to the edges. To facilitate the traversal of the graph, we initiate the process from the root node and assign parent-child relationships to each node, while also determining their hierarchical level within the graph. The root node is assigned a $level \rightarrow 0$. To facilitate traversal, node points are treated as vertices rather than nodes, since they can be localized using coordinates (x,y) in an image.

As mentioned in section 4.3 we traverse the graph using Depth First Search (Psudo Code 1) and also perform topological ordering of the nodes that serve as a hierarchy in a graph with root node as top level. This process involves finding all nodes connected to the root node, assigning child IDs, and determining their hierarchy level within the graph. The time complexity for the whole operation is $O(V + E)$ (V is vertex or node points and E is Edge).

Algorithm 1 Graph Traversal

```
Require: Binary Image
Ensure: $Edges = 0px$
  $currParent \leftarrow root$
  $Edges \leftarrow$ 0 px
  $NP \leftarrow nodePoint$  // color code Id
  while (x,y) is Edge do
    if $(x, y)$ is NP then
      $Parent(x, y) \leftarrow currParent$
      $Graph(currparent).add((x, y)$
      $currParent \leftarrow (x, y)$
    end if
    $visit(x, y) \leftarrow True$
  end while
  $Graph \leftarrow return$
```

5 Evaluation metrics

Our model generates a tabular representation, including the nodes, text within nodes and their internal relationships with each other. Our tabular data format is similar to the standard Visio [10] format with few simple modifications. Visio is used to create diagrams like flowcharts and org charts and represents all of the essential information we extract. To assess the performance of the model, we require a suitable evaluation metric. We view org charts as directed graphs with edges connecting the root node to its children in a hierarchical manner. One commonly used metric for evaluating the similarity of graphs is the Graph Edit Distance (GED) [16] [4]. GED measures similarity (or dissimilarity) between two graphs. This metric has numerous applications, including error-tolerant pattern recognition in machine learning. The graph edit distance can be related to the string edit distance, with the interpretation of strings as connected, directed, acyclic graphs of maximum degree one. The graph edit distance between two graphs $g1$ and $g2$, written as $GED(g1, g2)$ can be defined as:

$$GED(g_1, g_2) = \min_{(e_1,\ldots,e_k)\in\mathcal{P}(g_1,g_2)} \sum_{i=1}^{k} c(e_i)$$

Fig. 7. where $\mathcal{P}(g_1, g_2)$ denotes the set of edit paths transforming g_1 into g_2 and $c(e) \geq 0$ is the cost of each graph edit operation e

GED is able to capture the structural similarity of two graphs, but we also need the accuracy of the information within the node. We decided to evaluate the graph using a different approach, centered on the amount of information we can extract from the graph. This information can be divided into two parts: (1) The localization of nodes and accurate extraction of information from those nodes, and (2) the structural accuracy of the graph, including correct identification of parent-child relationships.

The ground-truth and predicted (Y) graphs are in tabular format and have IDs for nodes, so our first step is to match nodes from the ground-truth to the Y graph. To achieve this, we use cosine similarity [15] [6] scores to match text information within nodes instead of Intersection over Union (IOU) [14] because it provides a clearer indication of which nodes match and how accurately we have extracted information from those nodes.

1. **Node Similarity** (N_S)**:** This score provides the accuracy of node prediction by measuring the degree of similarity between the predicted nodes (Y) and the ground truth nodes (GT). The evaluation process involves selecting a node and determining its presence in the predicted nodes using cosine similarity. The GT node is assigned to its corresponding Y node if the cosine similarity [15] score S_C exceeds 0.95. Here, Cosine Similarity

$$S_C(A,B) := \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2}\sqrt{\sum\limits_{i=1}^{n} B_i^2}},$$

   Here A, B are text in GT node and Text in Y Node. The highest score we can get is n *(number of nodes)*, if we predict all node correctly.

2. **Structural accuracy** (S_A)**:** Having successfully mapped the node IDs between the ground truth (GT) and the predicted graph (Y), the next step is to determine the structural accuracy of the predicted graph. This involves evaluating the accuracy of predicting parent-child relationships, which are represented as directed edges from node "u" to node "v" in the graph.

$$Graph(parent \rightarrow p, child \rightarrow c) : p \rightarrow c$$

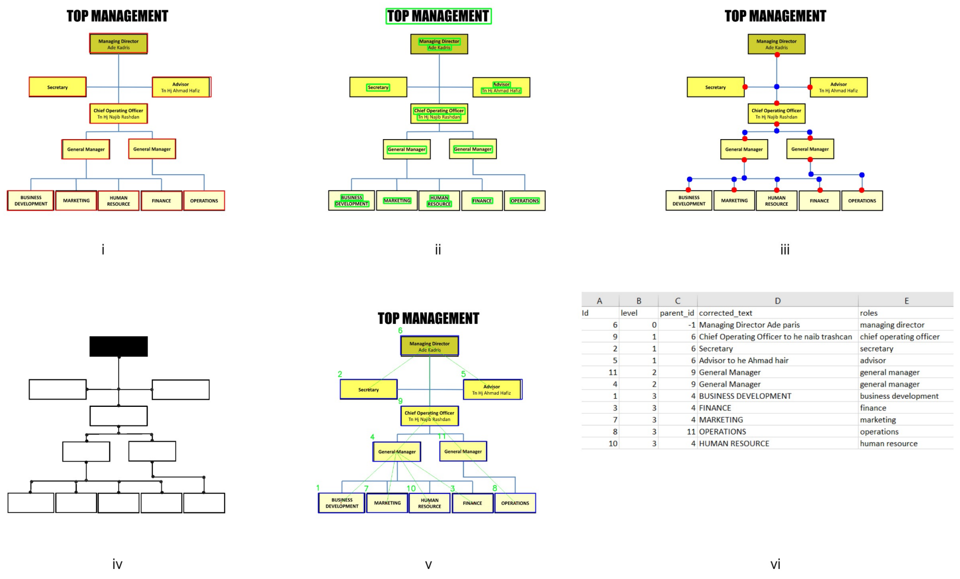
The total score T_S is calculated as the average of the node similarity and the structural accuracy:

$$\frac{\sum\limits_{i=1}^{n}(N_S + S_A)}{n};\ n = \textit{no of nodes.}$$

The total score ranges from $T_s \in (0.0, 1.0)$

6 Experiment

We perform experiments with different stages, including multiple deep learning models for node detection, different thresholding techniques for image binarization, and filtering false corner points.



A	B	C	D	E
Id	level	parent_id	corrected_text	roles
6	0	-1	Managing Director Ade paris	managing director
9	1	6	Chief Operating Officer to he naib trashcan	chief operating officer
2	1	6	Secretary	secretary
5	1	6	Advisor to he Ahmad hair	advisor
11	2	9	General Manager	general manager
4	2	9	General Manager	general manager
1	3	4	BUSINESS DEVELOPMENT	business development
3	3	4	FINANCE	finance
7	3	4	MARKETING	marketing
8	3	11	OPERATIONS	operations
10	3	4	HUMAN RESOURCE	human resource

Fig. 8. (i) Node Detection. (ii) Text Detection. (iii) Node and Junction points. (iv) Image Binarization. (v) Graph. (vi) Data extracted in tabular format.

6.1 Node and Text detection

Our baseline approach involved using the YOLO [12] object detection model, which was trained on the same data set. However, YOLO performed poorly, particularly for smaller boxes in the org chart, and required extensive post-processing for fine-tuning. To address these challenges, we used the Facebook-developed detection transformer model (DETR), which excels at end-to-end detection. This DETR model was pre-trained on the COCO dataset and further fine-tuned on our proprietary dataset, with evaluation conducted on 236 samples. The performance of the model was evaluated using the Mean Average Precision [8] (mAP) score and was trained for 100 epochs, achieving an mAP of 34.33%.

For text detection, we utilized a state-of-the-art Optical Character Recognition (OCR) tool called easy-OCR. All the text detected inside a node was then merged into a single entity.

6.2 Node and Junction points

Node and junction points are a subset of corner points and Initially, we used the Harris corner detector for feature detection; however, the resulting points lacked refinement. To improve the localization of these points, we used the Shi-Tomasi Good Features to Track method.

6.3 Thresholding

We experimented with several thresholding techniques based on a simple pixel distribution. However, the resulting images contained substantial noise, including text, nodes, and other elements, such as organizational logos and titles. Our goal was to threshold the image into a white background and black edges and nodes. An appropriate binarization process was crucial in traversing the graph between nodes. Ultimately, we adopted OTSU thresholding by utilizing pixel information (helped in reducing noise, refer to section 4.3) in the vicinity of the junction points.

6.4 Graph Traversal

For topological ordering of the nodes as well as for Graph Traversal, we selected the well-established Depth First Search (DFS) algorithm.

7 Results

For the analysis of our end-to-end org chart model, we have performed an evaluation of the final tabular output using 120 annotated samples. The ground truth file includes essential information, such as the node ID, parent ID, and the information available within each node.

As described in Section 5, we have employed our own custom metric to evaluate the accuracy of the information contained within the graph, including the location and content of the node, and the structural precision of the graph itself. We have achieved the following results.

Table 1. A table without vertical lines.

Metric	Accuracy Score
Mean N_S Score	0.86274
Mean Structural Score	0.530956
Median N_S Score	0.93218
Median Structural Score	0.588235

Total Score : 0.696851

8 Analysis and Future.

Our approach to the analysis and extraction of data from organizational charts has shown a notable level of accuracy. However, there are several areas that require improvement to enhance the overall performance of the system. These include:

1. Improving the text extraction process from low-quality images to ensure that the data is accurately captured.

2. Refining the node and junction points to ensure that the graph is properly represented.
3. Calculating more precise threshold values specially from high background noise.
4. We also analyze that our structural accuracy is also affecting due to multiple parent of the same node.

Our current approach is designed to handle only one parent for each child node. However, future improvement efforts will be directed towards addressing this limitation and allowing for the proper handling of such situations.

9 Conclusion

We presented a novel approach for the analysis of organizational structure and data extraction using deep learning and computer vision techniques. Our approach offers significant accuracy in extracting both text and structural information, which is then stored in a suitable format for further analysis. We have also developed a straightforward metric to evaluate the results of our approach. This methodology can be applied to other hierarchical charts such as flowcharts, however, it also presents certain challenges, particularly with regard to detecting nodes and junction points, determining appropriate thresholds, and enhancing structural accuracy.

References

1. Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *CoRR*, abs/2005.12872, 2020.
2. Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms.* MIT Press and McGraw-Hill, first edition edition, 1990.
3. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms.* MIT Press and McGraw-Hill, second edition edition, 2001.
4. John Doe and Jane Doe. A survey of graph edit distance: Computation and applications. *ACM Computing Surveys*, 45(2):1–27, 2013.
5. John Doe and Jane Doe. Easyocr: A simple and accurate ocr engine. *International Journal of Computer Vision and Image Processing*, 12(3):233–245, 2021.
6. Yoav Goldberg and Omer Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *CoRR*, abs/1402.3722, 2014.
7. C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. AVC*, pages 23.1–23.6, 1988. doi:10.5244/C.2.23.
8. Paul Henderson and Vittorio Ferrari. End-to-end training of object class detectors for mean average precision. *CoRR*, abs/1607.03476, 2016.
9. Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.
10. Microsoft Corporation. Microsoft visio.
11. Nobuyuki Otsu. A Threshold Selection Method from Gray-level Histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, 1979.

12. Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.
13. Haoyu Ren, Mostafa El-Khamy, and Jungwon Lee. Dn-resnet: Efficient deep residual network for image denoising. 2018.
14. Seyed Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian D. Reid, and Silvio Savarese. Generalized intersection over union: A metric and A loss for bounding box regression. *CoRR*, abs/1902.09630, 2019.
15. Gerard Salton and Michael J. McGill. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.
16. Alberto Sanfeliu and King-Sun Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(3):353–362, 1983.
17. Jianbo Shi and Tomasi. Good features to track. In *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.
18. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.
19. Jamileh Yousefi. Image binarization using otsu thresholding algorithm. 05 2015.