Learning Formatting Style Transfer and Structure Extraction for Spreadsheet Tables with a Hybrid Neural Network Architecture

Haoyu Dong
Microsoft Research
Beijing, China
hadong@microsoft.com

Jiong Yang
Xi'an Jiaotong University
Xi'an, China
yang1365340347@stu.xjtu.edu.cn

Shi Han
Microsoft Research
Beijing, China
shihan@microsoft.com

Dongmei Zhang
Microsoft Research
Beijing, China
dongmeiz@microsoft.com

ABSTRACT

Table formatting is a typical task for spreadsheet users to better exhibit table structures and data relationships. But quickly and effectively formatting tables is a challenge for users. Lots of manual operations are needed, especially for complex tables. In this paper, we propose techniques for table formatting style transfer, i.e., to automatically format a target table according to the style of a reference table. Considering the latent many-to-many mappings between table structures and formats, we propose CellNet, which is a novel end-to-end, multi-task model leveraging conditional Generative Adversarial Networks (cGANs) with three key components to (1) model and recognize table structures; (2) encode formatting styles; (3) learn and apply the latent mapping based on recognized table structure and encoded style, respectively. Moreover, we build up a spreadsheet table corpus containing 5,226 tables with high-quality formats and 784 tables with human-labeled structures. Our evaluation shows that CellNet is highly effective according to both quantitative metrics and human perception studies by comparing with heuristic-based and other learning-based methods.

CCS CONCEPTS

• **Information systems** → **Semi-structured data**; • **Computing methodologies** → **Neural networks**.

KEYWORDS

document intelligence; automatic formatting; deep learning

ACM Reference Format:
Haoyu Dong, Jiong Yang, Shi Han, and Dongmei Zhang. 2020. Learning Formatting Style Transfer and Structure Extraction for Spreadsheet Tables with a Hybrid Neural Network Architecture. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3340531.3412718

CIKM '20, October 19–23, 2020, Virtual Event, Ireland

ACM ISBN 978-1-4503-6859-9/20/10…$15.00
https://doi.org/10.1145/3340531.3412718

1 INTRODUCTION

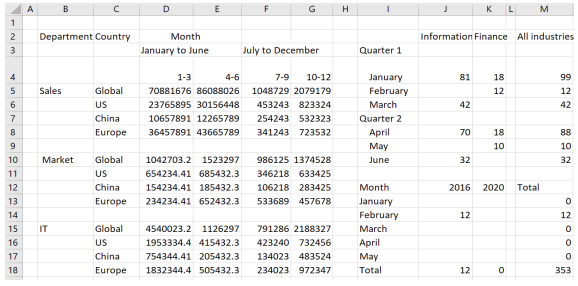
Spreadsheets is the most popular end-user development tools for data management, presentation, and analysis. Since spreadsheet tables are flexible with various structures, table formats such as border, font, alignment, and color, are created to help better exhibit table structures and data relationships. Take Figure 1(b)(c) for example, these well-formatted tables are much easier to read and understand for humans than the table in Figure 1(a) without any human-crafted formats.

However, table structures and data semantics are often complex. As a result, manual formatting of tables can be tedious and time-consuming. Automatic recommendation of table formats may help, but it would have difficulties when a user wants to enforce a consistent formatting style for multiple tables on the same spreadsheet, or wants to reuse a desirable formatting style from another table. Therefore, techniques are required to enable table formatting style transfer, i.e., to automatically format a target table according to the style of a reference table.

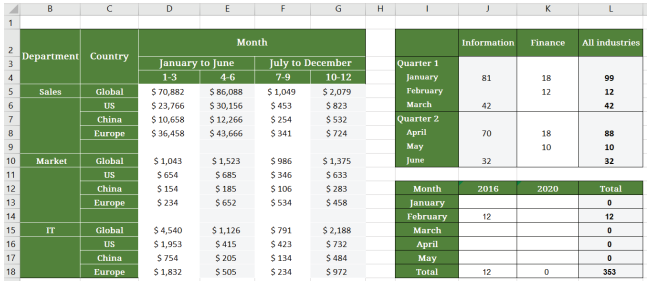
Table formats are highly dependent on table structures. For example, borders are usually used to shape data fields or data groups. And they are also dependent on data distribution and semantics, e.g., color/bold-font are usually used to highlight extreme/aggregation values. Therefore, we formulate the table style transfer task as a conditional generation task, i.e., generating the formats of a target table conditioning on both the structures and contents of it and a formatting style for reference. However, there lack key techniques to (1) model and recognize table structures; (2) encode formatting styles; (3) learn and apply the latent mappings between structure, style, and formats. In addition, table structure modeling concerns global relationships across all cells in the table, and the mappings from structures to formatting styles are latent and not one-to-one. These facts jointly challenge the applicability of supervised classification approaches.

In this paper, we propose CellNet, a hybrid data-driven model for table formatting style transfer. This model mainly contains two parts: the first part extracts table structures from table contents; the second part transfers the formatting style from a reference table to a target table conditioned on extracted table structures and the reference style. We conclude contributions in this paper as follows:

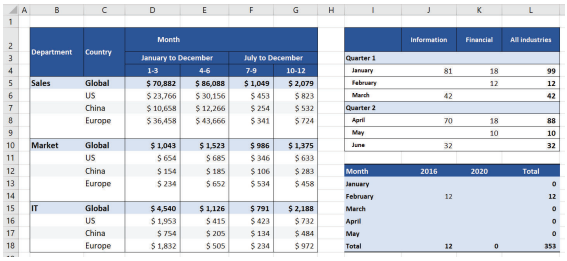
- We propose and formulate a new problem, formatting style transfer for spreadsheet tables. Since formatting generation



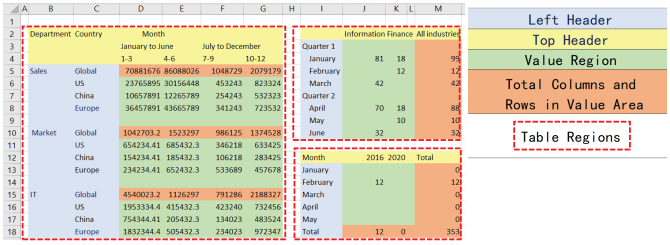
(a) A sheet with default formats.



(b) A sheet with human-crafted formats.



(c) A sheet with human-crafted formats.



(d) A sheet with human-labeled table structures.

Figure 1: An example of table formatting styles and the annotated table structure. (a) shows a spreadsheet consisting of three tables with default formats. (b) and (c) are two spreadsheets with the same table data but different formatting styles. (d) illustrates table structure annotations with intuitive coloring.

is highly dependent on table structures, we propose a joint learning framework, CellNet, to learn table structure extraction and table formatting generation simultaneously.

- For the task of table structure extraction, we propose a CNN-based model to detect table headers and total rows/columns. To enable supervised training, we build up a human-labeled dataset with broad coverage of diverse table structures.
- For the task of formatting style transfer, although paired data which can be directly consumed by supervised models are not available, large amounts of unpaired well-formatted tables are available on the web and can be easily obtained. Conditional Generative Adversarial Nets (cGANs) [7, 10, 13] is a promising framework to learn style transfer from unpaired data. While standard GANs [6] aim to learn a generative model that fits the natural data distribution, cGANs learn a conditional generative model to map data from one domain to the other, and have been successful in image style transfer [5, 14, 15]. So we propose a cGAN-based method for formatting style transfer conditioned on recognized table structures and the reference style.

Our experiments show the effectiveness of CellGAN. For table header prediction, it achieves 92.7% accuracy for top header prediction and 95.2% accuracy for left header prediction, outperforming all the baseline methods. CellNet achieves 95.2% precision and 82.4% recall for total row prediction, and it also achieves 93.3% precision and 88.6% recall for total column prediction. In the formatting style transfer task, both quantitative evaluations and human studies demonstrate the superiority of CellNet against other approaches.

2 PROBLEM STATEMENT

2.1 Table Structure Extraction

Table structure extraction is the task of recognizing structural component types inside a table. We identified five kinds of components for spreadsheet tables: top headers, left headers, value regions, total rows, and total columns. A **top header** is a rectangular range of cells that contains the labels of table columns and can be either flat or hierarchical. A **left header** is a rectangular range of cells that contains the labels of table rows. A **value region** is a rectangular range of cells that contains table data (i.e., the body of the table). Specifically, if a number in the value region is the sum of other numbers, then we call this number as a **total**. If the majority of numbers in a row are the sum of respective numbers in other rows, then we call this row as a **total row**. A **total column** is similar to a total row but is column-wise arranged. Total rows and total columns are commonly used in the spreadsheet table, e.g. total rows and columns are highlighted in orange in Figure 1(d). Moreover, totals are important indications for data groups, e.g. D5:G5, D10:G10 and D15:G15 in Figure 1(d) indicate three data groups consisting of Sales, Market and IT.

2.2 Formatting Style Transfer

Given a reference table A with style s_A and a target table B with style s_B, the style transfer task aims to generate formats f_{fake} for reference table A with similar style with s_B. But to the best of our knowledge, there are no existing research works on describing or defining table formatting styles. To capture various formatting

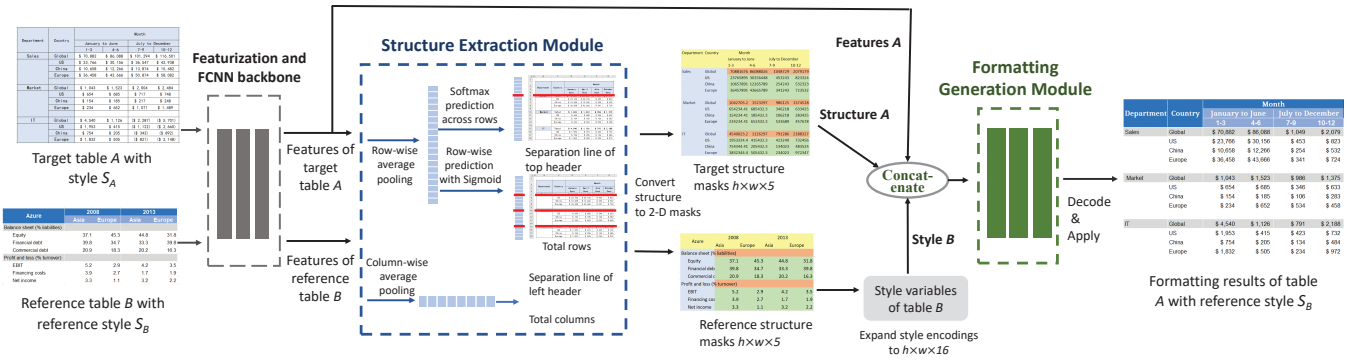


Figure 2: Network architecture in CellNet to transfer style from a target table B to the reference table A. The structure extraction module learns to recognize structural components and the formatting generator learns to generate formats based on style B.

styles in a more explicit way, we try to explore intuitive and meaningful style variables for controllable formatting generation. For example, we find an intuitive and quantitative variable to describe the density of cell borders in a region, and we call it border density. Here border density can be easily calculated by counting the proportion of the cells with borders in this region. Densities are also intuitive for other format types that can be controlled via a ratio-like variable, such as the portion of highlighted cells in the top header. Specifically, since fill color and font bold are commonly used to highlight the totals in a table, we directly use a boolean encoding to represent if a total row or a total column is highlighted with color or font bold. To this end, we list 16 intuitive and meaningful variables for style control as shown in Table 1. With the knowledge of table formats and table structure, all these variables can be obtained in a quantitative manner. Then we can use mean squared error (MSE) to quantify the similarity between the fake formatting style s_{fake} and the target formatting style s_{B}. In this paper, we adopt four commonly used format types, border, alignment, fill color, and font bold, to demonstrate the validity of our approach.

Table 1: Variables for formatting style encoding.

Description	Value
Border	
Proportion of horizontal borders in value region.	[0.0, 1.0]
Proportion of vertical borders in value region.	[0.0, 1.0]
Proportion of horizontal borders in top header.	[0.0, 1.0]
Proportion of vertical borders in top header.	[0.0, 1.0]
Proportion of horizontal borders in left header.	[0.0, 1.0]
Proportion of vertical borders in left header.	[0.0, 1.0]
Color (transparent and white are excluded)	
Proportion of colorized cells in value region.	[0.0, 1.0]
Proportion of colorized cells in top header.	[0.0, 1.0]
Proportion of colorized cells in left header.	[0.0, 1.0]
Total rows are colorized or not.	{0, 1}
Total columns are colorized or not.	{0, 1}
Font bold	
Proportion of cells with font bold in value region.	[0.0, 1.0]
Proportion of cells with font bold in top header.	[0.0, 1.0]
Proportion of cells with font bold in left header.	[0.0, 1.0]
Total rows are set to bold or not.	{0, 1}
Total columns are set to bold or not.	{0, 1}

3 METHOD

3.1 Datasets

The WebSheet dataset is a large-scale web-crawled spreadsheet corpus including 4,290,022 spreadsheet files[3]. We obtained our training and testing data via the following procedure: First, spreadsheets in WebSheet are grouped by their HTTP domains and selected with the top 12 domains with high formatting quality and broad coverage of diverse table structures. Second, all tables in selected spreadsheets are extracted by a recent table detection approach, TableSense[3]. Third, 15% of the spreadsheet tables of each domain are selected for human labeling according to the structure definitions in Section 2.1. Finally, a final corpus containing 5,226 tables with high-quality formats and 784 tables with structure labeling can be obtained. 70% of the tables are randomly selected as the training set, and the remaining 30% are used for testing. Table 2 shows the statistics for each dataset on table structure and formatting.

3.2 Cell Featurization

Cells do not have a canonical representation in the spreadsheet, but contain rich information such as data types, data formats, cell formats, formulas, etc. To capture effective representations to initiate end-to-end model learning, we devise a featurization scheme. Each cell is encoded by a 21-dimensional vector to capture cell string, data type, merge, and formula, whereas the formats are encoded by a 6-dimensional vector to capture border, fill color, alignment, and font bold. And we also devise a 124-dimensional vector to capture row-level or column-level features, e.g., we devise a row-level feature that describes whether the formulas in the current row have the same relative references. Then we expand both the row-level and column-level features to the table size of $h \times w$. Based on our

Table 2: Comparison results of datasets on table structure and formatting.

Dataset	Average rows per table	Average columns per table	Average total rows per table	Average total columns per table	Average rows per top header	Average columns per left header	Proportion of cells with vertical border	Proportion of cells with horizontal border	Proportion of cells with fill color	Proportion of cells with font bold
census.gov	46.0	10.6	2.3	0.5	2.6	1.1	0.97	0.99	0.22	0.73
edr.state.fl.us	50.4	14.4	6.0	0.9	1.8	2.7	0.99	1.00	0.98	1.00
nces.ed.gov	48.8	13.2	1.1	0.2	2.7	1.1	0.54	0.97	0.07	0.74
nsf.gov	33.4	10.1	2.1	0.2	2.1	1.0	0.46	1.00	0.10	0.32
saus	39.0	12.0	1.1	0.2	4.2	1.1	0.84	0.89	0.01	0.77
ucr.fbi.gov	40.1	10.8	2.2	1.1	1.5	1.6	0.10	1.00	0.00	0.84
juntadeandalucia	31.7	10.4	1.3	0.7	1.6	1.1	0.03	0.99	0.00	0.93
censusindia.gov	53.4	15.3	0.0	0.7	3.7	0.3	0.78	1.00	0.01	0.17
contraloria.gob	38.7	9.9	2.0	0.7	3.2	1.3	1.00	1.00	0.43	0.91
dane.gov.co	38.0	11.4	0.8	0.3	2.1	1.4	0.18	1.00	0.60	0.93
irs.gov	43.1	10.4	2.2	0.4	3.5	1.5	0.90	0.99	0.02	0.85
nyc.gov	41.4	8.72	0.3	0.1	1.3	0.7	0.97	1.00	0.56	0.95

proposed featurization, an $h \times w \times 151$ input tensor d and an $h \times w \times 6$ output tensor f can be extracted from a table with $h \times w$ cells. Table 3 provides some representative features for table data and formats, and the complete feature schema can be found in Appendix A.

Table 3: Table data and formats featurization.

Name	Description	Value
Value string		
Log length	Log length of the string.	Float
Alpha prop	Proportion of the letters in the string.	[0.0, 1.0]
Has total keyword	If this cell contains "total" keyword.	[0.0, 1.0]
Merged Cell		
Merged with top	If the cell is merged with top neighbor.	{0, 1}
Merged with left	If the cell is merged with left neighbor.	{0, 1}
Data type		
Is number type	If the cell value is displayed as number.	{0, 1}
Is date type	If the cell value is displayed as date.	{0, 1}
Formula		
Has formula	If the cell value has formula.	{0, 1}
Has SUM	If contains the SUM in formula.	{0, 1}
Formats		
Horizontal border	If horizontal border of the cell exists.	{0, 1}
Vertical border	If vertical border of the cell exists.	{0, 1}
Horizontal alignment	The horizontal alignment (left, center, right).	{0, 1, 2}
Vertical alignment	The vertical alignment (top, middle, bottom).	{0, 1, 2}
Fill color	If color exists (except transparent and white).	{0, 1}
Font bold	If font bold of the cell is added.	{0, 1}

3.3 Network Architecture

Figure 2 presents our network architecture for semantic structure extraction and formatting style transfer. Given reference table A and target table B, CellNet first extracts structures of table A and table B. Second, the style s_B of table B can be calculated based on the structure and formats of table B. Then CellNet generates formats for table A conditioned on the reference formatting style s_B. Three key modules in CellNet are listed as follows:

CNN Backbone: The input tensor d_A of table A and d_B of table B encode cell-level features. Here we employ a fully convolutional neural network (FCNN) [9] to learn representations r_A and r_B from input tensor d_A and d_B, respectively. This CNN backbone is trained jointly by structure extraction and formatting style transfer.

Table Structure Extraction: Given input tensor d_A and representations r_A of table A, the structure extraction module learns to predict the top separation line between the top header and the data region, the left separation line between the left header and the data region, and total rows/columns. Row-wise and column-wise average pooling are adopted to squeeze feature maps in horizontal and vertical directions, respectively. Since one table may have multiple total rows, each row uses a sigmoid function for classification (similar for total column prediction). While one table only has one top header, a softmax function is adopted to predict a single separation line across rows (similar for left header prediction). The prediction results are then converted to structure masks m_A with 5 channels as shown in Figure 1(d), each channel representing a specific type of structure component.

Formatting Style Transfer: we adopt a cGAN-based method to learn the mapping from style to formatting, such that the generated formatting is indistinguishable from the real-world formatting. Given input tensor d_A, feature representations r_A, structure masks m_A and reference style s_B, the formatting style transfer module learns to generate formats for the input table A in conditional settings. As shown in Figure 2, to allow low-level spreadsheet features to be effectively used to generate the output formats, the generator G adopts a "u-net" architecture [12]. In our setting, although m_A is fed to the formatting module, the loss of formatting module does not backprop to the structure extraction module.

3.4 Objective Functions

The objective function of table structure extraction $\mathcal{L}_{\text{structure}}$ can be defined straightforwardly by averaging the cross-entropy loss of row/column-wise predictions for total rows/columns and header separation lines, while the objective function of formatting style transfer is hard to define since the absence of paired data in our formatting style transfer task. In our dataset, one table is only formatted with one formatting style, so given a target table A with style s_A and a reference table B with style s_B, there does not exist the corresponding formats of table A with style s_B to serve as ground truth. So it's challenging for traditional discriminative models that need paired training data and explicitly defined loss functions. To address this challenge, we define 16 intuitive and meaningful variables shown in Table 1 to formulate styles in a quantitative way and propose Style Consistency Loss $\mathcal{L}_{\text{style}}$ to describe the style consistency between the generated formatting style and the reference

formatting style as follows:

$$\mathcal{L}_{\text{style}} = \sum_{i=1}^{16} (s_{\text{fake}_i} - s_{\text{B}_i})^2. \tag{1}$$

Moreover, table formats are not independent cell-by-cell class labels but are well organized on a two-dimensional spreadsheet. Generative Adversarial Nets (GANs) and conditional GANs (cGANs) show that using a trained network as a loss function (discriminator) can synthesize highly structured outputs (e.g. natural images) [6, 10, 14]. Thus we utilize cGAN's framework by adopting a discriminator D to guide the generated formats to look "real", which is trained simply to classify whether the pair of table input and formats $(\boldsymbol{d}, \boldsymbol{f})$ look "real" or "fake". The objective function is given by:

$$\begin{aligned}\mathcal{L}_{\text{cGAN}}(G, D) =& \mathbb{E}_{(\boldsymbol{d},\boldsymbol{f})}[\log D(\boldsymbol{d}, \boldsymbol{f})] + \\ & \mathbb{E}_{\boldsymbol{d}}[\log(1 - D(\boldsymbol{d}, G(\boldsymbol{d})))],\end{aligned} \tag{2}$$

The final objective function combines the losses above, and the whole model is trained simultaneously in an end-to-end way:

$$\mathcal{L} = \mathcal{L}_{\text{structure}} + \lambda \mathcal{L}_{\text{cGAN}} + \gamma \mathcal{L}_{\text{style}}, \tag{3}$$

where λ and γ control the weights of these three terms.

4 EXPERIMENTS

4.1 Implementation Details

Inference takes on average 24.2 milliseconds per table on a V100 GPU. λ and γ are set to 5.0 and 2.0, respectively. The entire model is trained end-to-end using an Adam optimizer. The learning rate is initialized to 0.0001.

Generator in CellNet Let Ca×b-k denote a convolutional layer with k a×b filters and stride 1. Accordingly, Da×b-k denotes convolutions down-sampled by a factor of 2, whereas convolutions up-sampled by a factor of 2 are denoted as Ua×b-k. RRCP-k denotes a residual block that contains 5×3 convolutional layers with k filters on both layers. Then the generator consists of:

- Encoder: C1×1-64, C7×3-64, C7×3-64, D8×2-128, D8×3-256, D8×3-512, D8×3-1024, RRCP-1024.
- Decoder: U8×3-1024, U8×2-512, U8×3-256,U8×2-128,U8×2-5

Discriminator in CellNet The discriminator adopts multi-scale PatchGAN [14] and consists of: C1×1-64, C5×3-64, D7×3-128, D5×3-128, C3×1-512, D7×3-512, C5×3-512, C5×3-1.

4.2 Top/Left Header Recognition Evaluations

We invest a rule-based method for header recognition as well as widely-used machine learning models such as Gradient Boosting Decision Tree (GBDT):

Rule-based baseline we implement a header identification method based on type-and-format mixed inference as a baseline.

GBDT baseline we train a GBDT model to predict the number of rows in the top header and the number of columns in the left header based on the feature schema introduced in Appendix A.

Evaluation results are shown in Table 4. CellNet achieves significant accuracy gains over baseline models.

Table 4: Comparison results of header recognition.

%	Top header accuracy	Left header accuracy
Rule-based	63.1	77.2
GBDT	84.6	93.9
CellNet	**92.7**	**95.2**

Table 5: Comparison results of total recognition.

%	Total rows		Total columns	
	Precision	Recall	Precision	Recall
Rule-based method	26.8	48.8	43.7	68.8
GBDT	89.4	79.1	89.6	**89.2**
CellNet	**95.2**	**82.4**	**93.3**	88.6

4.3 Total Rows/Columns Recognition Evaluations

Since there lack research works on total rows and total columns recognition, we first investigate a rule-based method as well as a widely-used machine learning model, GBDT, on recognizing total columns and total rows given a table.

Rule-based baseline we implement heuristics based on the feature schema introduced in Appendix A. The key feature we used is "If Approx. Row Sum", which is listed in combined row-wise features. This feature describes if a data row can be approximately summed up by other continuous data rows. But this algorithm is brittle when a table has blank cells or round operations to data.

GBDT baseline we train a GBDT model to predict the total rows and columns based on the same feature schema introduced in Appendix A. For total rows, the GBDT model gives 0/1 prediction for each table row. And similar to total column prediction.

The comparison results of precision and recall on the testing set are shown in Table 5. CellNet shows significant gains over both the rule-based method and GBDT in total row/column prediction. We attribute the performance gains to the power of deep convolutional models for capturing various spatial relationships of cells.

4.4 Formatting Style Transfer Evaluations

To the best of our knowledge, there's no research work on spreadsheet table formatting style transfer. We investigate a rule-based method as well as a variant of CellNet for this task.

Rule-based baseline We implement a rule-based method to format tables by consolidating heuristics based on our dataset. First, we parse the header hierarchy to a top header tree and a left header tree based on the merged cells in the top header and indent level in the left header. To be specific, we build a hierarchical relationship in the top header between a merged cell and the following cells under it. It is similar in the left header for the cell and its subsequent cells with more indents. According to the hierarchy, we format appropriate levels of this hierarchical from top to bottom, whose proportion of cells is closest to the specified format density in the corresponding region. Additionally, the hierarchy of headers reflects the data groups in the value region, which indirectly decides the formats in value region. For example, we format horizontal and

vertical borders in value region depending on left and top header hierarchy, respectively, and we format color and bold based on the most matched level in the top and left hierarchies.

CellNet (w/o cGAN loss) CellNet (w/o cGAN loss) is a variant of CellNet which directly replaces $\mathcal{L}_{\text{cGAN}}$ by a supervised loss $\mathcal{L}_{\text{supervision}}$ without transferring formatting. CellNet (w/o cGAN loss) tries to optimize MSE between the generated formats f_{fake} and the real format f_{A} of table A without using adversary loss.

$$\mathcal{L}_{\text{supervision}} = \frac{1}{wh} \sum_{i=0,j=0}^{i<h,j<w} (f_{\text{fake}_{i,j}} - f_{\text{A}_{i,j}})^2, \tag{4}$$

where h and w denote the height and width of table A.

4.4.1 Human Perceptual Study. Table formatting is mainly applied to help users understand table structures and data relationships intuitively. Thus the subjective perception of humans is important to evaluate the quality of table formatting. So we conduct a human perceptual study by employing 8 Excel professionals from Speechocean [1] to evaluate the formatting generation results of various methods. For each table, the generated formats and its original real-world formats form a pair for human comparison. A professional has unlimited time to inspect a pair and decide "better quality", "worse quality", or "a tie for comparable qualities" according to:

- **Integrity:** Whether the formatting looks complete without missing/redundant formats.
- **Effectiveness:** Whether the formatting reflects table structure well for easy look-up.
- **Harmony:** Whether the formatting visually matches the table data in harmony.

In this experiment, all tables in the testing set are provided for human evaluation. For each reference table in the testing set, we randomly select another table in the testing set as the reference table B, then we use the models described above to generate formats for the target table A conditioned on s_{B} of table B.

As shown in Table 6, all types of formats generated by CellNet are rated much higher than those produced by alternative approaches. For borders, 82.3% of the table format results obtained by CellNet are considered to be no worse than the real-world data by human users, showing significant improvement over 51.8% achieved by the rule-based baseline. For alignment generation, the rule-based baseline achieves 97.3% and CellNet even achieves 97.5%, and both numbers are high. But CellNet(w/o cGAN loss) only achieves 72.4%, which is much lower than other methods. A reasonable explanation is that human evaluators are not sensitive to the overall alignment direction while they are sensitive to local outliers. Our case studies show that by training only with the structure and style loss, CellNet(w/o cGAN loss) produces much more local defects compared with other methods. Thus we conclude that the loss $\mathcal{L}_{\text{cGAN}}$ is capable of enhancing the quality of generated formats.

4.4.2 Table-Level Accuracy. In this section, we propose a novel table-Level accuracy (TLA) metric, which measures the degree of exact matching between generated and real formats as follows:

[1] http://en.speechocean.com/

Table 6: Human evaluation results. These numbers indicate the proportion of generated formatting that is no worse than (better or comparable) real-world formatting.

%	Color	Bold	Border	Alignment
Rule-based	85.5	87.8	49.0	97.3
CellNet(w/o cGAN loss)	88.5	78.9	51.8	72.4
CellNet	**91.2**	**95.7**	**82.3**	**97.5**

Table 7: TLA results of formatting generation for target table A conditioned on its own style s_{A}.

%	Color	Bold	Border	Alignment
Rule-based	85.1	32.5	23.2	36.0
CellNet(w/o cGAN loss)	87.2	64.6	58.4	70.5
CellNet	**89.2**	**75.1**	**67.8**	**78.3**

$$\text{TLA} = \sum_{n=1}^{N} |\boldsymbol{f}^n_{\text{fake}} == \boldsymbol{f}^n_{\text{A}}|/N, \tag{5}$$

where N is the total number of test tables, and "==" returns 1 if the generated formats $\boldsymbol{f}^n_{\text{fake}}$ of table n and the corresponding real formats $\boldsymbol{f}^n_{\text{A}}$ are exactly the same in the whole table region.

But as discussed in Section 3.4, due to the absence of ground truth formats for the task of formatting style transfer, we cannot directly evaluate the TLA metric in the task of formatting style transfer. Here we devised an experiment that adapts the formatting style transfer task to a formatting generation task. For each reference table A in the testing set, instead of generating formats conditioned on the reference table A and reference style s_{B}, models are used to generate formats conditioned on reference table A and reference style s_{A}. Then this task is adapted to a pure supervised task, and the real-world formats of table A can be directly used as ground truth. To avoid ground truth leaking from input features, we exclude formatting information from the input feature d. This experiment can be utilized to evaluate the abilities of models to generate real-world likely formats.

As shown in Table 7, CellNet outperforms all comparison methods. For border generation, the average TLA is 67.8%, which significantly outperforms the rule-based baselines of 23.2% and the CellNet(w/o cGAN loss) baseline of 58.4%.

4.4.3 Style Consistency Evaluations of Formatting Style Transfer. To evaluate the ability of methods to keep the consistency of the generated style s_{fake} and the reference style s_{B}, we evaluate MSE between s_{fake} and s_{B}. Our experiments show that the average MSE achieved by CellNet is 0.0058, much lower than 0.0068 of the rule-based method. CellNet (w/o cGAN loss) is not evaluated here because it can not produce reasonable formatting patterns in this task without adversary loss. And as discussed before, CellNet achieves much better results in human evaluations and quantitative metrics for formatting quality, so we can conclude that CellNet outperforms baselines both in style consistency and formatting quality.

5 RELATED WORK

Spreadsheet Table Structure Extraction and Formatting Style Transfer As far as we know, this is the first study on learning spreadsheet table formats conditioned on table structures. Nor can we find any features or tools in Excel for intelligent formatting. The most related feature is conditional formatting, which allows users to manually define conditions for conducting batch mode coloring or visual augmentation, but is not applicable for adaptive style transfer. As for table structure extraction, [1, 11] aim to recognize tables and analyse their structures, but only target on web tables. [2, 8] propose interactive ways for extracting relational metadata from spreadsheet tables. [4] aims to identify expandable groups in spreadsheets. CellNet is the first method to learn table structure extraction and formatting generation in a joint way.

6 DEPLOYMENT

The proposed techniques for table style transfer is still at its incubation stage towards products, but its promising effectiveness has been recognized and appreciated via internal deployments for trial in Microsoft. The internal trial is invite-only, opened to multiple product teams in Office involving cross-disciplinary employees including Program Managers, Designers, Engineers, and Engineering Managers. Since the first deployment and invited trials in March 2020, we have received good amount of positive feedback, which verifies the practical utility of our techniques.

The deployment mainly consists of two parts, the front-end interface and the back-end service. The front-end part is responsible for user interaction, table reading and format applying, while the machine learning models are hosted in the back-end service to extract table structures and transfer the formatting style from one to another. We deploy the front-end part and publish an Office add-in in Microsoft in a network sharing way[2], and both a local service and an online service are deployed on the back-end side. A real demo video for this Office add-in can be found on our website[3].

For the front-end part, we adopt Office JavaScript API[4] to read Excel tables and apply formats. It supports Excel in both Office and Office 365 and has nearly real-time performance. As shown in the demo video, by clicking the "Extract" button after selecting a reference table and then clicking "Apply" after selecting a target table, one can easily transfer the style from a reference table to a target table. Moreover, to enforce a consistent formatting style for multiple tables on the same document, a user can apply a selected formatting style to the whole document via simple clicks.

For the back-end part, we wrap our key technologies as a service: (1) extracting table structures; (2) encoding table formatting styles regarding different table structures; (3) adaptively decoding and applying a reference style to a table. By simply calling (3) for each table accordingly, our add-in can support the scenario of making multiple tables in a document with a consistent style.

[2]https://docs.microsoft.com/en-us/office/dev/add-ins/publish/publish

[3]https://www.microsoft.com/en-us/research/publication/learning-formatting-style-transfer-and-structure-extraction-for-spreadsheet-tables-with-a-hybrid-neural-network-architecture/

[4]https://docs.microsoft.com/en-us/office/dev/add-ins/develop/understanding-the-javascript-api-for-office

7 DISCUSSION AND CONCLUSION

In this paper, we propose formatting style transfer for spreadsheet tables. First, as introduced in Section 1, table formats depend on table structures. To enable high-accuracy data-driven methods for table structure extraction, we build up a spreadsheet table corpus with high-quality formats and structure labeling. Second, since there lacks a widely adopted learning framework in the spreadsheet domain, especially for generation tasks, we propose CellNet to learn and apply the latent mapping from table data to table formats based on recognized table structures and encoded styles. In the future, we plan to build a unified intelligent experience across Office applications for automatic table formatting.

8 ACKNOWLEDGEMENTS

We would like to thank Ran Jia and Xiao Lv for assistance with table data extraction, table detection, and structure analysis; Chaochao Tang, Shujia Jiang, Yue Xiao, Qiwei Meng, Biao Cheng, Changxu Wang, and Bin Wang for designing and deploying the Office add-in in this paper. Lei Zhang and Chao Hu for help defining different table structures and exploring mothods for table style transfer.

REFERENCES

[1] Marco D Adelfio and Hanan Samet. 2013. Schema extraction for tabular data on the web. *Proceedings of the VLDB Endowment* 6, 6 (2013), 421–432.

[2] Zhe Chen and Michael Cafarella. 2014. Integrating spreadsheet data via accurate and low-effort extraction. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1126–1135.

[3] Haoyu Dong, Shijie Liu, Shi Han, Zhouyu Fu, and Dongmei Zhang. 2019. Tablesense: Spreadsheet table detection with convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 69–76.

[4] Wensheng Dou, Shi Han, Liang Xu, Dongmei Zhang, and Jun Wei. 2018. Expandable group identification in spreadsheets. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*. 498–508.

[5] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2016. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2414–2423.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[7] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. 2017. Learning to discover cross-domain relations with generative adversarial networks. *arXiv preprint arXiv:1703.05192* (2017).

[8] Elvis Koci, Dana Kuban, Nico Luettig, Dominik Olwig, Maik Thiele, Julius Gonsior, Wolfgang Lehner, and Oscar Romero. 2019. XLIndy: interactive recognition and information extraction in spreadsheets. In *Proceedings of the ACM Symposium on Document Engineering 2019*. 1–4.

[9] Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3431–3440.

[10] Mehdi Mirza and Simon Osindero. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784* (2014).

[11] Kyosuke Nishida, Kugatsu Sadamitsu, Ryuichiro Higashinaka, and Yoshihiro Matsuo. 2017. Understanding the semantic structures of tables with a hybrid deep neural network architecture. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 234–241.

[13] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. Adversarial discriminative domain adaptation. In *Computer Vision and Pattern Recognition (CVPR)*, Vol. 1. 4.

[14] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. 2018. High-resolution image synthesis and semantic manipulation with conditional gans. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Vol. 1. 5.

[15] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*. 2223–2232.

A FEATURE LIST APPENDIX

A.1 Basic Cell-wise Features

Table 8 mainly describes basic featurization of cells, covering nine categories such as cell string, merged cell, data type, and so on.

Table 8: Cell-wise Features

Category	Name	Description	Value
Cell String	Log Length	Log of length of the string.	Float
	Log length of Space	Log length of the prefix spaces in string.	Float
	Alpha Prop	Proportion of the letters in the string.	[0.0, 1.0]
	Number Prop	Proportion of digits in the string.	[0.0, 1.0]
	Keyword Total	If the keyword 'total' exists in the string.	{0, 1}
	Keyword Sum All	If the keyword 'sum' or 'all' exists in the string.	{0, 1}
Merge	Merged	If the cell is merged.	{0, 1}
	Merged With Top	If the cell is merged with top neighbor.	{0, 1}
	Merged With Bottom	If the cell is merged with bottom neighbor.	{0, 1}
	Merged With Left	If the cell is merged with left neighbor.	{0, 1}
	Merged With Right	If the cell is merged with right neighbor.	{0, 1}
Data Type	Number Type	If the cell value is viewed as number.	{0, 1}
	Date Type	If the cell value is viewed as date.	{0, 1}
	Same With Right	If the format string is consistent with that of right neighbor.	{0, 1}
	Same With Below	If the format string is consistent with that of the neighbor below.	{0, 1}
Formula	Has Formula	If the cell has formula.	{0, 1}
	Refers Current Row	If the formula refers any cell in the current row explicitly.	{0, 1}
	Refers Current Column	If the formula refers any cell in the current column explicitly.	{0, 1}
	Contains Sum	If the formula uses the function 'sum'.	{0, 1}
	Same With Right	If the formula in relative form is consistent with that of right neighbor.	{0, 1}
	Same With Below	If the formula in relative form is consistent with that of the neighbor below.	{0, 1}
Indent	Indent Quantity	Numbers of indents in this cell.	Integer
Border	Horizontal Border	If the cell has top border.	{0, 1}
	Vertical Border	If the cell has left border.	{0, 1}
Alignment	Horizontal Alignment	Horizontal alignment of the cell (left, center, right)	{0, 1, 2}
	Vertical Alignment	Vertical alignment of the cell (bottom, center, top)	{0, 1, 2}
Color	Colored	If the color of the cell is neither white nor transparent.	{0, 1}
	R Value	R value of the RGB color.	[0.0, 1.0]
	G Value	G value of the RGB color.	[0.0, 1.0]
	B Value	B value of the RGB color.	[0.0, 1.0]
Bold	Bold Font	If bold font is applied.	{0, 1}

A.2 Row/column-wise Features

Combined features are derived from basic features, and are divided into two parts, one for rows and the other for columns. Row combined features are listed in Table 9 , and similar for column features.

A.3 Features for Total Row/column Detection

Total features mostly serve for the recognition of total rows or columns, which also have two parts for rows and columns, respectively. In Table 10 we only shows a single feature group with a specific absolute or relative error threshold, and different absolute or relativ thresholds can form different feature groups. In our complete total feature list, there are twelve groups in which six groups serve for row total features with six different absolute or relative error thresholds and the other six groups serve for columns with similar patterns. In Table 10, we leave out the group features of columns for simplicity.

Table 9: Combined Row-wise Features

Category	Name	Description	Value
Cell String	Occurance Number	Number of occurances of the cell string in the current row.	[0.0, 1.0]
	Distance To Next	Distance to the next cell string occurance in the current row.	Integer
	String Repeat Rank	The rank of current occurance number in the current row.	Integer
	All Empty	If all the strings and formulas in the current row are empty.	{0, 1}
	Non-empty Prop	Proportion of cells with non-empty strings in the current row.	[0.0, 1.0]
	Keyword Total	If there exists a cell has keyword 'total' in the current row.	{0, 1}
Data Type	Number Prop	Proportion of cells with data type of number in the current row.	[0.0, 1.0]
	Date Prop	Proportion of cells with data type of date in the current row.	[0.0, 1.0]
	Incons. Prop	Proportion of data type inconsistency between cells in the current row and corresponding cells in the next row.	[0.0, 1.0]
Border	Horizontal Prop	Proportion of cells with horizontal(top) border in the current row.	[0.0, 1.0]
	Vertical Prop	Proportion of cells with vertical(left) border in the current row.	[0.0, 1.0]
Color	Not White Prop	Proportion of cells with non-white and non-transparent color in the current row.	[0.0, 1.0]
Bold	Bold Prop	Proportion of cells with bold font in the current row.	[0.0, 1.0]
Alignment	Vertical Top Prop	Proportion of cells with vertically top alignment in the current row.	[0.0, 1.0]
	Vertical Center Prop	Proportion of cells with vertically center alignment in the current row.	[0.0, 1.0]
	Vertical Bottom Prop	Proportion of cells with vertically bottom alignment in the current row.	[0.0, 1.0]
Formula	Formula Prop	Proportion of cells with formulas in the current row.	[0.0, 1.0]

Table 10: Features for Detecting Total Rows

Category	Name	Description	Value
In-row	Non-zero Value Prop	Proportion of cells with non-zero numerical value in the current row.	[0.0, 1.0]
Cross-row	If Approx. Row Sum	If a row-continuous region exists, in which more than 60% of values summed up by rows are approximately equal with corresponding cell values in the current row within an absolute/relative error.	{0, 1}
	If Approx. Cell Sum	If the numerical value of the cell is approximately equal with the sum of cells in the corresponding region within an absolute/relative error.	{0, 1}
	Approx. Equal Prop	Proportion of cells in the current row, whose numerical values are approximately equal with the sum of cells in the corresponding region within an absolute/relative error.	[0.0, 1.0]
	Region Multi-rows	If the region has multiple rows.	{0, 1}
	Region Distance	The shortest distance between the corresponding region and the current row.	Integer
	Region Adjacent	If there exists the corresponding region is closely adjacent to the current row.	{0, 1}
	Region Location	Location of the corresponding region respecting to the current row (top, down).	{0, 1}