A Appendix

A.1 Implementation Details

The hyperparameters we used for pre-training and fine-tuning downstream tasks are shown in Table 1. In general, our hyperparameter settings are similar to the one used in Deformable DETR [11]. Here, “base” configurations are those common for all experiments. It is worth mentioning that we use the box refinement design proposed in [11] which we find helpful for bounding box prediction.

For “pre-training”, we use a three-stage pre-training. That is, in the first stage, we pre-train on 1M IIT-CDIP samples [6] for 20 epochs. Then, we pre-train on 5M samples for 5 epochs. In the final stage, we pre-train on 11M samples (full dataset) for 2 epochs.

“receipt parsing”, “entity labeling” and “entity linking” show the settings we used to obtain the numbers we reported in Table 1 and 2 of the main manuscript. For “entity extraction”, since we follow the existing work to address this task, the hyperparameter settings are similar to the one used in DETR [11]. Here, “base” configurations are those common for all experiments. It is worth mentioning that we use the box refinement design proposed in [11] which we find helpful for bounding box prediction.

In the experiments, the default settings (that match Table 1) are marked in gray.

Table 1: Implementation settings. Here, “LR” stands for learning rate. “EE” stands for entity extraction. “EL” stands for entity linking. “CE” stands for cross-entropy.

<table>
<thead>
<tr>
<th>config type</th>
<th>config name</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>optimizer</td>
<td>AdamW [7]</td>
<td></td>
</tr>
<tr>
<td>base LR</td>
<td>2e−4</td>
<td></td>
</tr>
<tr>
<td>cnn LR</td>
<td>2e−5</td>
<td></td>
</tr>
<tr>
<td>base</td>
<td>language encoder L.R.</td>
<td>1e−5</td>
</tr>
<tr>
<td></td>
<td>weight decay</td>
<td>1e−4</td>
</tr>
<tr>
<td></td>
<td>LR schedule</td>
<td>step</td>
</tr>
<tr>
<td></td>
<td>box refinement [11]</td>
<td>yes</td>
</tr>
<tr>
<td></td>
<td>batch size</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>epochs</td>
<td>20, 5, 2</td>
</tr>
<tr>
<td></td>
<td>training samples</td>
<td>1M, 5M, 11M</td>
</tr>
<tr>
<td></td>
<td>LR drop step size</td>
<td>20</td>
</tr>
<tr>
<td>pre-training</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>batch size</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>epochs</td>
<td>200</td>
</tr>
<tr>
<td></td>
<td>LR drop step size</td>
<td>160</td>
</tr>
<tr>
<td>receipt</td>
<td>EE loss weight</td>
<td>5.0</td>
</tr>
<tr>
<td>parsing</td>
<td>EL loss weight</td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>anchor word</td>
<td>first+last</td>
</tr>
<tr>
<td></td>
<td>primary anchor</td>
<td>name</td>
</tr>
<tr>
<td>entity</td>
<td>batch size</td>
<td>8</td>
</tr>
<tr>
<td>labeling</td>
<td>epochs</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>LR drop step size</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>CE loss weight</td>
<td>2.0</td>
</tr>
<tr>
<td>entity</td>
<td>batch size</td>
<td>8</td>
</tr>
<tr>
<td>linking</td>
<td>epochs</td>
<td>200</td>
</tr>
<tr>
<td></td>
<td>LR drop step size</td>
<td>160</td>
</tr>
<tr>
<td></td>
<td>EE loss weight</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>EL loss weight</td>
<td>10.0</td>
</tr>
</tbody>
</table>

Next, we show some additional experiments to further demonstrate our model’s property under different settings. In the experiments, the default settings (that match Table 1) are marked in gray.

**Comparison of using anchor words from different line-item fields as primary anchors.** From Table 2, we can see that using the anchor word of “name” gives the best result. This is because in the test set, all the line-items contain this field. So, it is reliable to use this field as the primary anchor for entity linking. For the other fields, their performances...
Comparison of using predicted text and ground truth text as inputs. The existing works [4, 2, 9, 1] use ground truth text as the input in the experiments. We also follow the same way for fair comparison. However, it would be interesting to see how the models work if predicted text (from an OCR system) is used. In particular, using ground truth text as input is not in favor of vision only approaches such as Donut [5]. Table 3 shows the results of the comparison. In this experiment, we use an in-house OCR system which has comparable performances with the state-of-the-art OCR solutions (e.g., those from Azure, GCP or AWS). As we can see, there is a performance drop when we switch from using ground truth text to predicted text. However, compared with the vision only solution Donut, we are still noticeably better. This indicates the importance of having language inputs.

Model performance at different pre-training stages. In Table 4, each stage is based on the pre-trained model from its previous stage. For example, stage 2 initializes the model using the weights pre-trained from stage 1. As we can see, when more data is used, the model’s performance continues improving on the FUNSD entity extraction task.

Visualization of language-conditioned cross-attention. We further verify the behavior of this cross-attention mechanism by visualizing the cross-attention matrices. The cross-attention results are extracted from the cross-attention module of each decoder layer, i.e., we check the cross-attention between the language inputs and the language-conditioned queries. As we can see in Fig 2, the attention weights are high on the diagonal of the attention matrices. This shows that we successfully established this one-to-one mapping between the queries (and thus decoder outputs) and language tokens.
**Additional visualizations.** Fig 3-6 show additional comparisons of three structured information extraction formulations. Fig 7 show two receipts parsing failure cases. Fig 8-11 show additional pre-training outputs using our proposed masked detection modeling task.

**References**


Figure 3: CORD receipt parsing results on receipt_00002 sample. Each result consists of the visualization of model predictions, and the parsing outputs. (a) IOB tagging visualizes the predicted tags of OCR words. (b) SPADE visualizes the decoded graph, and arrows between words indicate that the words are linked in the same entity. (c) Ours visualizes the predicted anchor words and their bounding boxes. For simplicity of the visualization, entity-linking results are not visualized in here. For the parsing outputs, green/red text means the predicted text matches/does not match ground truth. Strikethrough text means the ground truth text is missed from prediction. Best view in color and zoom-in for details of the visualization. For this example, both IOB tagging and SPADE recognized the second row of the line-item as an individual line-item. DocTr understands the line-item better by recognizing it as a single line-item.
IOB Tagging

```
[ { "menu.nm": "Goblin's Mace" },
  { "menu.price": "25,000" },
  { "menu.cnt": "1" } ],
[ { "menu.nm": "Cheese Croquette" },
  { "menu.price": "25,000" },
  { "menu.cnt": "1" } ],
[ { "menu.nm": "Chili Pepper Croquette " },
  { "menu.price": "0" } ],
[ { "sub_total.discount_price": "14,000" },
  { "sub_total.subtotal_price": "91,000" }] 
```

(b) SPADE

```
[ { "menu.nm": "Goblin's Mace" },
  { "menu.price": "25,000" },
  { "menu.cnt": "1" } ],
[ { "menu.nm": "Mozzarella Hot Dog" },
  { "menu.price": "38,000" },
  { "menu.cnt": "2" } ],
[ { "menu.nm": "Chili Pepper" },
  { "menu.price": "25,000" },
  { "menu.cnt": "1" } ],
[ { "sub_total.discount_price": "0" },
  { "sub_total.subtotal_price": "91,000" }] 
```

(c) Ours

```
[ { "menu.nm": "Goblin's Mace" },
  { "menu.price": "25,000" } ],
[ { "menu.nm": "Mozzarella Hot Dog" },
  { "menu.price": "38,000" } ],
[ { "menu.nm": "Chili Pepper" },
  { "menu.price": "25,000" },
  { "menu.cnt": "1" } ],
[ { "sub_total.discount_price": "0" },
  { "sub_total.subtotal_price": "91,000" }] 
```

Figure 4: CORD receipt parsing results on receipt_00017 sample. Each result consists of the visualization of model predictions, and the parsing outputs. (a) IOB tagging visualizes the predicted tags of OCR words. (b) SPADE visualizes the decoded graph, and arrows between words indicate that the words are linked in the same entity. (c) Ours visualizes the predicted anchor words and their bounding boxes. For simplicity of the visualization, entity-linking results are not visualized in here. For the parsing outputs, green/red text means the predicted text matches/does not match ground truth. Strikethrough text means the ground truth text is missed from prediction. Best view in color and zoom-in for details of the visualization. This is a challenging sample with several line-items. Some line-items have single-line name, and some have two-line names. As we can see, SPADE totally failed in this case, with a wrong graph for the line-item names. IOB tagging and DocTr are better and detected names mostly correctly.
Figure 5: CORD receipt parsing results on receipt_00058 sample. Each result consists of the visualization of model predictions, and the parsing outputs. (a) IOB tagging visualizes the predicted tags of OCR words. (b) SPADE visualizes the decoded graph, and arrows between words indicate that the words are linked in the same entity. (c) Ours visualizes the predicted anchor words and their bounding boxes. For simplicity of the visualization, entity-linking results are not visualized in here. For the parsing outputs, green/red text means the predicted text matches/does not match ground truth. Strikethrough text means the ground truth text is missed from prediction. Best view in color and zoom-in for details of the visualization. In this example, “CASH” is a subtotal. But it is right below the line-item. Thus, both “IOB Tagging” and “SPADE” missed this detection.

Figure 6: CORD receipt parsing results on receipt_00078 sample. Each result consists of the visualization of model predictions, and the parsing outputs. (a) IOB tagging visualizes the predicted tags of OCR words. (b) SPADE visualizes the decoded graph, and arrows between words indicate that the words are linked in the same entity. (c) Ours visualizes the predicted anchor words and their bounding boxes. For simplicity of the visualization, entity-linking results are not visualized in here. For the parsing outputs, green/red text means the predicted text matches/does not match ground truth. Strikethrough text means the ground truth text is missed from prediction. Best view in color and zoom-in for details of the visualization.
Figure 7: CORD receipt parsing failure cases. We visualize the predicted anchor words, their bounding boxes, and anchor word associations (yellow arrows).
Figure 8: Example pre-training predictions on FUNSD sample 0060024314. For inputs, we visualize masked word boxes, and their text is replaced by [MASK]. For predictions, we visualize the predicted word boxes of the masked inputs. Under each box prediction, we also visualize its corresponding word token predictions.
Figure 9: Example pre-training predictions on FUNSD sample 01073843. For inputs, we visualize masked word boxes, and their text is replace by [MASK]. For predictions, we visualize the predicted word boxes of the masked inputs. Under each box prediction, we also visualize its corresponding word token predictions.
Figure 10: Example pre-training predictions on FUNSD sample 660978. For inputs, we visualize masked word boxes, and their text is replace by [MASK]. For predictions, we visualize the predicted word boxes of the masked inputs. Under each box prediction, we also visualize its corresponding word token predictions.
Figure 11: Example pre-training predictions on FUNSD sample 91914407. For inputs, we visualize masked word boxes, and their text is replace by [MASK]. For predictions, we visualize the predicted word boxes of the masked inputs. Under each box prediction, we also visualize its corresponding word token predictions.