PLAtE: A Large-scale Dataset for List Page Web Extraction

Aidan San[*]
University of Virginia
aws9xm@virginia.edu

Yuan Zhuang[*]
University of Utah
yuan.zhuang@utah.edu

Jan Bakus[†]
Amazon

Colin Lockard
Amazon
clockard@amazon.com

David Ciemiewicz
Amazon
ciemo@amazon.com

Sandeep Atluri
Amazon
satluri@amazon.com

Yangfeng Ji
University of Virginia
yangfeng@virginia.edu

Kevin Small
Amazon
smakevin@amazon.com

Heba Elfardy
Amazon
helfardy@amazon.com

Abstract

Recently, neural models have been leveraged to significantly improve the performance of information extraction from semi-structured websites. However, a barrier for continued progress is the small number of datasets large enough to train these models. In this work, we introduce the PLAtE (**P**ages of **L**ists **At**tribute **E**xtraction) benchmark dataset as a challenging new web extraction task. PLAtE focuses on shopping data, specifically extractions from product review pages with multiple items encompassing the tasks of: (1) finding product-list segmentation boundaries and (2) extracting attributes for each product. PLAtE is composed of $52,898$ items collected from $6,694$ pages and $156,014$ attributes, making it the first large-scale list page web extraction dataset. We use a multi-stage approach to collect and annotate the dataset and adapt three state-of-the-art web extraction models to the two tasks comparing their strengths and weaknesses both quantitatively and qualitatively.

1 Introduction

Semi-structured data extraction, i.e., web extraction, is the task of extracting data found in templated text fields from HTML pages. Once extracted, the structured data can be utilized in various downstream tasks such as information retrieval, recommendation, and question answering.

While recent work has shown the potential of neural approaches for web extraction (Lin et al., 2020; Zhou et al., 2021; Li et al., 2021), there are very few publicly available large-scale datasets suitable for training and evaluation of these approaches, limiting progress in this area. Additionally, most existing datasets (Hao et al., 2011; Hotti et al., 2021) focus on one subset of the problem; namely *detail page extraction*. In this paper, we introduce the PLAtE (**P**ages of **L**ists **At**tribute **E**xtraction) dataset[1], that specifically targets the task of *list page* extraction and focuses on product review pages in the shopping vertical, i.e., multi-product review pages.

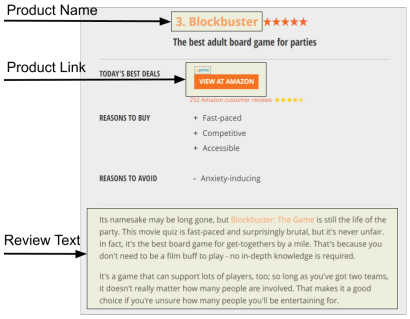

Figure 1: A single item (i.e. product) from a list page for a board game (from gamesradar.com).

To elaborate, item pages can be broadly categorized into two classes: *detail pages* and *list pages*. Detail pages provide detailed information about a single item. List pages comprise a list of items with abridged detail, organized under a single theme, e.g. "best board games". This organization facilitates direct comparison of each item and allows for the extracted data to be easily integrated into recommender, question answering, or dialogue systems powering digital assistants (Linden et al., 2003; Gupta et al., 2019; Zhang et al., 2018). Extracted product data can be utilized by both content creators (publishers) as well as customers looking to make purchase decisions. [2]

Because PLAtE is built from list (multi-item) pages, we can evaluate two tasks: segmentation

[*] The work was completed while Aidan and Yuan were interning at Amazon.

[†] Jan left Amazon, but the work was completed while he was working at Amazon.

[1] PLAtE will be publicly available at https://github.com/amazon-science/plate

[2] We provide an example of a list page and a detail page in the appendix.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 5: Industry Track, pages 284–294

and attribute extraction. In the segmentation task, the model should determine the boundaries of each product, i.e. where the information for each product begins and ends. In the attribute extraction task, each node in the HTML DOM tree can be assigned any number of three labels: product name, product review text, and product link. The dataset is comprised of $52,898$ products from $6,694$ pages split into: training, development, and held-out test sets. To evaluate the dataset, we adapt two state-of-the-art neural web extraction models; MarkupLM (Li et al., 2021) and DOM-LM (Deng et al., 2022), as well as explore a text-based Transformer model, RoBERTa (Liu et al., 2019). We achieve an F1-score of 0.787 and 0.735 for segmentation and attribute extraction tasks respectively. We evaluate the potential of multi-task learning to improve performance and find that multi-task learning improves recall but slightly decreases precision, for both tasks. To summarize, our contributions are: (1) creating the first large-scale *list page* web extraction dataset, (2) adapting state-of-the-art neural web extraction approaches to the segmentation and attribute extraction tasks, and (3) qualitatively and quantitatively comparing the performance of different models on the two presented tasks; segmentation and attribute extraction.

2 Related Work

The vast majority of previous web extraction datasets (e.g., SWDE (Hao et al., 2011), the Klarna Product Page Dataset (Hotti et al., 2021), and WDC (Petrovski et al., 2016)) are composed of single item pages. Multiple item or list page datasets are much less common. Zhu et al. (2006) created a dataset of 771 list pages, while Dhillon et al. (2011) created a small dataset (BSCM) of about 30 pages from 4 verticals: Books, Seminars, CS Faculty and MLConfs. Furche et al. (2012) built a dataset of 431 pages from two verticals: UK used car dealer websites collected from a used car aggregator website and UK real-estate websites collected from the yellow pages. To the best of our knowledge, PLAtE is the largest multi-item web extraction dataset. Additionally, most previous web extraction datasets assume a single DOM node has at most a single label. However, this assumption does not hold true in many product pages. For example, for many products, the product name's text-span is also a link to the product. Our dataset does not have this assumption; we allow a node to have multiple labels. Finally, most existing list page datasets are not publicly available. Table 1 compares the different web extraction datasets.

From the methods side, web extraction has first been tackled using wrapper induction methods that create a set of rules (wrappers) to transform unstructured input into structured output (Kushmerick, 1997; Furche et al., 2014; Zheng et al., 2007; Azir and Ahmad, 2017; Gulhane et al., 2011; Carlson and Schafer, 2008; Furche et al., 2012). Recently, a number of advances have been made by utilizing neural-based approaches to construct a representation for each HTML node for the extraction task (Lockard et al., 2020; Lin et al., 2020; Zhou et al., 2021; Li et al., 2021; Deng et al., 2022; Xie et al., 2021).

Other tasks related to semi-structured information extraction include boilerplate removal (Leonhardt et al., 2020), extraction from HTML tables (Cafarella et al., 2018; Deng et al., 2020; Wang et al., 2021; Herzig et al., 2020), and segmenting pages, e.g., using clustering followed by string alignment (Álvarez et al., 2008), optimization based on divide and conquer (Bing et al., 2013), and Hierarchical Conditional Random Fields (CRFs) (Zhu et al., 2006).

3 PLAtE Benchmark

In this work, we tackle two tasks: *(1) segmentation*, i.e., identifying the boundaries of the individual products in a given page and *(2) attribute extraction*, i.e., identifying the individual attributes for each identified product. For each given product, we extract the following three attributes: (1) *product name*: This refers to the name of the product, e.g. "iPhone 11", (2) *review text*: This is generally a high-level review or general description of the product, and (3) *purchase link*: a link (or button) to a merchant's website (e.g., Amazon, Ebay). We commonly see generic text such as "Buy Here", the name of the merchant such as "Amazon", or the name of the product. Similar to prior work, we perform classification on the leaf nodes; i.e., only nodes that contain text are passed to the classification models.

3.1 PLAtE Construction Process

To construct PLAtE, we started with 270M documents from the large, publicly available web crawl Common Crawl.[3] We then filtered down to $6,694$

[3] https://commoncrawl.org/

Name	Pages/Vert.	Sites/Vert.	Records/Vert.	Tot. Pages	Year	Mult. Item?
SWDE (Hao et al., 2011)	4,405-20,000	10	4,405-20,000	124,291	2011	✗
WDC (Petrovski et al., 2016)	576	?	576	576	2016	✗
Klarna (Hotti et al., 2021)	51,701	8,175	51,701	51,701	2019	✗
LDST (Zhu et al., 2006)	771	?	$\sim$ 8600	771	2006	✓
(Dhillon et al., 2011)	5-15	5-8	$\sim$ 400	$\sim$ 30	2011	✓
AMBER (Furche et al., 2012)	150-281	**100-150**	1608-2785	431	2012	✓
PLAtE	**6,694**	43	**52,898**	**6,694**	**2020**	✓

Table 1: Comparison of existing web extraction datasets against PLAtE. PLAtE has the greatest number of records per vertical, and the freshest HTML content. Additionally, it has an order of magnitude larger number of records than any of the other multiple item datasets. "?" Means the information is not available in the paper. The multi-item dataset with the highest statistic is bolded and the overall dataset with the highest statistic is underlined.

candidate pages from 43 internet websites by (1) removing duplicate URLs and non-English pages, (2) filtering out non-multi-product pages using a linear classifier with word embedding and TF-IDF features as well as keywords-based heuristics (e.g., "best", "compare", etc.), (3) selecting top/popular websites using the Tranco List (Pochat et al., 2019), (4) selecting sites with the highest number of list pages, and (5) filtering out pages with inappropriate (i.e. sensitive or offensive) content.

After selecting the candidate pages, we performed the initial segmentation and attribute extraction by using CSS selector rules.[4] Two expert annotators used a custom annotation tool to annotate a representative page from each site by selecting a CSS selector for each attribute. The annotated CSS selectors were then used to extract the attributes from the rest of the pages from the same site. Multiple rounds of quality checks were performed in order to ensure the quality of the final selector rules.

The final step in creating PLAtE used Amazon Mechanical Turk annotations in order to remove any errors introduced by the rule-based extraction step. For the Mechanical Turk annotation task, we presented a web page to the annotators. We first asked the annotators a set of page-level questions to ensure that the web page is a valid multi-product review page. We then asked the annotators to verify that a piece of highlighted text within the web page should be extracted as an attribute and asked them to indicate if any text of the attribute of interest was not highlighted, i.e., was not captured by the rules.[5] Overall, 20% of workers that attempted the qualification task were qualified, resulting in 77 annotators. To identify spammers, we blocked any annotator that spent less than 20 seconds on average per annotation task. Finally, to minimize annotation costs while ensuring high-quality evaluation and development data, we used one annotation per task for the training set and three annotations per task for the development and test sets. Majority vote was used to aggregate the annotations from the development and held-out test sets.

To build the final dataset, we split the data such that the training, development, and held-out test sets have approximately the same distribution in terms of number of products and pages. Moreover, we ensured that sites from the same website family, e.g., *thespruce.com* and *thespuceeats.com*, appear in the same split. Table 2 shows a sample PLAtE annotation, while Table 3 shows statistics of the dataset.

Site	theinventory.com
URL	theinventory.com/best-iphone ...
Product Index	4
Attr. Name	Product Name
Num. Extracted	1
XPath	/html/body/div[3]/ ... /span/a/strong
Text	['AirPods Pro']

Table 2: An example PLAtE annotation.

Split	# Sites	# Pages	# Products	# Attrs
Train	28	4,202	35,383	103,731
Dev	5	655	6,038	18,019
Test	10	1,837	11,477	34,264
All	43	6,694	52,898	156,014

Table 3: Statistics of the train, development, and test sets.

[4]CSS selector rules are patterns composed of HTML tag names and HTML class names used to select one or more HTML elements (e.g., `[.product-slide p]`)

[5]We only qualified annotators from English speaking countries that completed at least 100 prior annotation tasks with an acceptance rate of 95% or more, and who passed a custom qualification task.

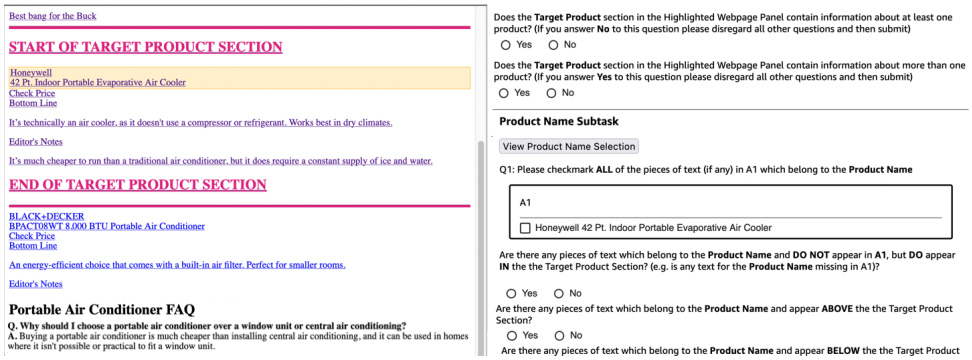

Figure 2: Sample Mechanical Turk task. In the left panel, the text-span "Honeywell42. Pt Indoor Portable Evaporative Air Cooler" is highlighted as the extraction of interest. In the right panel, a checkbox denotes whether the highlighted text-span falls under the named attribute, i.e., "Product Name". The annotator is also asked to determine whether the left panel is a valid product and if any named attributes are missing.

Attribute	Tag	Text	Crowd Annotations	Gold
Review Text	<p>	And those are our recommendations for the best mattresses!...	[True, False, False]	False
Product Link	<a>	Tempur-Pedic	[True, True, False]	True
Review Text	<p>	And those are our recommendations for best outdoor grills...	[True, True, False]	False
Product Link	<a>	Original Sprout's miracle detangler	[True, False, False]	True

Table 4: Annotator disagreements on the page `https://www.wisebread.com/the-5-best-mattresses`.

Attribute	Missing(%)	Valid (%)	Fleiss-Kappa
Product Name	2.7	97.0	0.596
Review Text	2.3	85.1	0.728
Product Link	6.0	96.7	0.560

Table 5: Annotation statistics.

3.2 Dataset Analysis

We manually analyzed PLAtE annotations to assess their quality. In the annotation task, as seen in Figure 2, the first two annotation questions were designed to filter out cases of mislabeled or malformed snippets. Specifically, the first question asked whether the annotation snippet contained at least one product while the second question asked if the snippet contained more than one product. Annotators indicated that 99.6% of snippets contained at least one product and that only 0.8% contained more than one product, indicating that the vast majority of tasks sent to annotators were valid product snippets. The next set of annotation questions are meant to identify if any annotations were missing from the presented snippet. Between 3% and 6% of the extracted attributes were missing some text spans. Finally, the last set of questions asked the annotators to select the text-spans that matched an attribute from a set of check-boxes (all text-spans matching the CSS rule). Overall, more than 85% of the text-spans were correctly matched to the corresponding attribute. We also calculated the inter-annotator agreement using Fleiss-Kappa (Fleiss and Cohen, 1973) to measure the quality of the annotation guidelines and found that the inter-annotator agreement is moderate for "*Product Name*" and "*Product Link*" and substantial for "*Review Text*". For "*Product Link*", some product name spans were also product link spans, causing confusion among the raters. The annotation statistics are shown in Table 5. Percentage of missing attributes is relatively low indicating that our CSS selector rules have high recall. Additionally, percentage of valid extractions is quite high especially for *Product Name* and *Product Link* indicating that our CSS selector rules have high precision. Table 4 shows examples of annotator disagreement. The third example is challenging because the text is about outdoor grills in general, and not a particular outdoor grill product, so it should be annotated as False. In the fourth example, the annotators likely missed the correct label (*True*), because the link does not follow the standard format of "*Buy Now*" or a retailer's name such as "*Amazon*".

4 Models

We evaluate the performance of three recent neural models: RoBERTa (Liu et al., 2019), DOM-LM (Deng et al., 2022) and MarkupLM (Li et al., 2021) on PLAtE.

Model	Attribute Extraction			Segmentation				
	Test P	Test R	Test F1	Test P	Test R	Test F1	Test ARI	Test NMI
RoBERTa	0.843	0.652	0.735	0.692	0.665	0.678	0.693	0.744
DOMLM	0.815	0.655	0.726	0.718	0.728	0.722	0.716	0.764
MarkupLM	0.839	0.620	0.711	0.769	0.805	0.787	0.771	0.870

Table 6: Performance of the different models on the segmentation and attribute extraction tasks. For segmentation, MarkupLM has the best performance while for attribute extraction, DOM-LM outperforms RoBERTa on Recall, but RoBERTa overall has the highest F1.

RoBERTa is a Transformer-based (Vaswani et al., 2017) model pre-trained with natural language texts sourced from the BookCorpus (Zhu et al., 2015) and Wikipedia. The pre-training task is masked language modeling. In our experiments, the input to RoBERTa is a sequence of text tokens from the DOM tree. Consequently, it does not utilize other types of information from the DOM tree such as XPaths or HTML tags.

DOM-LM is designed to generate contextualized representations for HTML documents. It uses RoBERTa as the base encoder and is pre-trained over the SWDE dataset (Hao et al., 2011) with masked language modeling over the text tokens as well as the DOM tree nodes. To encode a DOM tree, DOM-LM first slices the tree into a set of subtrees such that important tree-level context is kept in each subtree. Then each node is represented by its tag, text, class/id attributes, as well as a positional matrix based on its position in the DOM tree.

MarkupLM is another RoBERTa-based model. The input to MarkupLM is a node represented by both an XPath embedding and its text. The XPath embedding is created by embedding each tag and subscript in the XPath separately and then concatenating and passing them through a FFN layer. The model was pre-trained on three tasks: (1) masked markup language modeling, (2) node relation prediction, and (3) title page matching using 24 million pages from Common Crawl.

For the segmentation task, we label each of the nodes as *begin* (B) to denote the first text-span of a product, or *other* (O) for the rest of the nodes. We then apply a softmax layer to the logits and train with cross-entropy loss. For the attribute extraction task, the model predicts the attribute labels from the logits using a multi-label sigmoid layer that was trained with binary cross-entropy loss. As a multi-label classification task, the model can assign each node with any subset of the labels or no label. In both tasks, we begin with one of the three pre-trained models and then fine-tune on the training set of PLAtE.

5 Experiments

For both segmentation and attribute extraction tasks, we report the precision, recall, and F1-score. Results are macro-averaged over the different classes. In addition, for segmentation, we report clustering metrics following (Bing et al., 2014), where the attributes in the same segment are considered to be in the same cluster. In our dataset, when looking at two adjacent products in a page, there is a single node which we label as a "segmentation boundary". In reality, there can be multiple nodes which appear between two adjacent products and split the products apart. If multiple valid segmentation boundaries for a product are possible, F1-score will penalize the model for picking any segmentation boundary which is not labelled as such. On the other hand, the clustering metrics provide a relaxation which checks that product attributes are assigned to the correct product. For clustering metrics, we report adjusted rand index (ARI) (Hubert and Arabie, 1985) and normalized mutual information (NMI) (Strehl and Ghosh, 2002) where a higher number indicates better performance. [6,7]

Overall, we observe that MarkupLM performs well on the segmentation task yielding scores of 0.787, and 0.771 and 0.870 for F1, ARI and NMI respectively while DOM-LM performs worse on this task with scores of 0.722 and 0.716 and 0.764 for F1, ARI and NMI respectively. This can likely be attributed to the fact that MarkupLM was pre-trained on the task of relation prediction between nodes. Identifying if two nodes have a parent-child, sibling, or ancestor-descendant relation could help the model distinguish nodes within the same product from nodes of different products, and conse-

[6] ARI ranges from -0.5 to 1 and NMI ranges from 0 to 1.

[7] We average all results for each of the two tasks over three runs from different random seeds.

Model	Attribute	P	R	F1
RoBERTa	Product Name	0.858	0.645	0.737
	Review Text	0.922	0.721	0.808
	Product Link	0.750	0.590	0.661
DOM-LM	Product Name	0.843	0.672	0.747
	Review Text	0.863	0.695	0.769
	Product Link	0.741	0.599	0.662
MarkupLM	Product Name	0.885	0.621	0.730
	Review Text	0.822	0.671	0.737
	Product Link	0.808	0.569	0.667

Table 7: Precision, recall, and F1-score for all models on the attribute extraction task by attribute type.

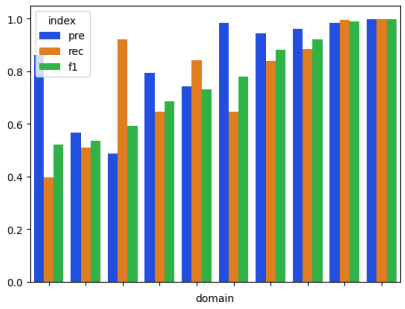

Figure 3: Precision, recall, and F1 of MarkupLM's attribute extraction scores grouped by site. We observe a high variance in scores, due to differing HTML structure between different sites.

quently identify the product boundaries better.

The attribute extraction task shows a different trend from segmentation where RoBERTa performs the best, outperforming MarkupLM and DOM-LM. We look into the reason behind this in Section 6. Detailed results for both tasks are presented in Table 6 while precision, recall, and F1-score for the attribute extraction task broken down by attribute are shown in Table 7. We find that product link extraction performs worst while product review text performs best. For product link, this difficulty can be attributed to the diverse nature of product links which can take the form of a *product name*, or the *price of the product* e.g., a hyperlink with the text $10 or a *button* (e.g. with the text "Buy Now"). For review text, the higher performance is not surprising given that it normally has a more consistent style than product links.

To better understand the different factors affecting the performance of the attribute extraction task, we (1) break down the scores by site in our test set to see whether some sites are more challenging than others, (2) analyze the relationship between the number of products in a page and attribute extraction performance (3) explore whether the index of the product (i.e., where it appears in the page) affects the performance, (4) study the effect of adding more pages (versus sites) to the training data, and (5) explore whether a multi-task model that jointly tackles both segmentation and attribute extraction tasks can yield better performance through utilizing the complementary signals between both tasks.

As can be seen in Figure 3, extraction performance varies greatly by site. We suspect that this is due to the diverse page layouts and styles of different sites. When grouping pages by number of products, we find that for all three models, as the number of products increases, the F1-score improves. We believe this is due to the increased ease of the model to learn patterns within a page given more examples in a page. When grouping the scores by product index, we find that product index does not have a significant impact on the performance of attribute extraction; i.e. performance is relatively uniform across different locations in the page.[8]

To study the value of annotating more sites compared to annotating more pages, we sampled two subsets of pages from our original training set. We collected all pages from 14 randomly sampled sites from the training set for a total of 1514 pages, then sampled 1514 random pages across all sites in the training set. The 14 sites achieved an F1=0.595 compared to sampling from all sites achieving an F1=0.681. From this, we can conclude there is indeed value in annotating a greater sites instead of simply annotating more pages from the same site. We suspect the model is better able to generalize to the test set, when provided with a more diverse training set of different sites. Finally, we look into whether multi-task learning improves model performance on PLAtE. To this end, we train a MarkupLM model with a shared encoder, but distinct loss functions and output layers for the segmentation and attribute extraction tasks.[9] We find that multi-task learning improves the recall of both tasks – as shown in Table 8 – but slightly harms precision. F1-score performance increases by 0.023 for the segmentation task but goes down for the attribute extraction task due to the decrease in precision.

[8]Figures showing the detailed results of these experiments are provided in the appendix.

[9]We use a weighted sum of the loss functions for both tasks and determine the weights empirically based on the performance on the development set.

Task	Setting	P	R	F1
Attribute Extraction	No MTL	0.839	0.620	0.711
	MTL (weight=0.5)	0.803	0.628	0.703
Segmentation	No MTL	0.769	0.805	0.787
	MTL (weight=0.25)	0.768	0.859	0.810

Table 8: Precision, recall, and F1 for Multitask Learning (MTL) for the MarkupLM model.

6 Discussion

Contrary to our expectations, RoBERTa outperformed DOM-LM, a model specially designed to model webpages, on the attribute extraction task. As can be seen in Table 7, RoBERTa mainly outperforms DOM-LM in review text precision. We analyze 20 examples where DOM-LM makes a false positive review text error and RoBERTa does not. We find in 95% of the examples, the misclassified node is outside of the main review content, and in 80% of the examples the misclassified node is a <p> tag. This indicates that DOM-LM's structural encoder is likely over-fitting to the HTML tag and disregarding the text content, hence is less able to generalize to unseen HTML structures.

We perform additional analysis of the outputs from the MarkupLM model, to identify areas for improvement for the attribute extraction task. Specifically, we sampled up to 5 false positive (FP) and 5 false negative error examples for each attribute from each site in the test set, and ended up with a total of 257 examples.[10] For product review text, we find that many of the false negatives are very short textspans such as "*These*", "*as well*", and "*require*" representing a single text leaf node within a larger paragraph. We suspect that the model has not been trained with enough examples of varied text length. In the future, we could also consider training the model to classify the parent nodes representing a whole paragraph instead of classifying at the leaf node level. For false positives, we find 4 examples of user-written comments (as opposed to reviewer-written text). While these do not represent the official review text, they are semantically similar to the official review text hence are easily confused by the model. One such example is a user's comment mentioning that "*the design and functionality of these cookers is top-notch*" which has a similar style to text which could have been written by a reviewer. For product link, 17 false positives are hyperlinks that link to a homepage rather than to a specific product which indicates that the model is over-fitting to the <a> tag. Training the model using contrastive learning with positive and negative product link <a> tag examples should help with this case.

Next we explore the effect of threshold-tuning on the performance of attribute extraction. In the presence of an oracle that specifies whether or not a node should be tagged with an attribute, we can use the $argmax$ of the logits of the different attribute classes in order to guarantee that the node is tagged with one of the three attributes. Based on the very high agreement between the $argmax$ and the actual label for product name (89.4%) and product review text (91.8%), it is clear that some of the challenges in tagging nodes in PLAtE stem from needing to determine whether or not a node should be tagged with any attribute, on top of determining what the correct attribute tag is.

Finally, we analyze the errors in the segmentation task, collecting all pages with an ARI and NMI < 0.5 within test set. We find that in 97% of these pages the number of gold segmentation boundaries is higher than the number of predicted segmentation boundaries. This means that when ARI and NMI are very low, the model is failing to split the page into enough segments. To improve performance, we could consider utilizing a structured prediction to model segmentation interdependencies e.g. the model could explicitly model that it is unlikely that two segmentation boundaries appear directly next to one another.

7 Conclusion

In this work, we introduce PLAtE, the first large-scale list page dataset for web extraction. We describe our methodology in creating the dataset, from pre-processing to crowd-sourced annotation collection. We evaluate the performance of three strong web extraction baselines and achieve an F1-score of 0.787 and 0.735 on the segmentation and attribute extraction tasks respectively. While we focus on shopping domain due to its importance to several downstream applications, in the future we intend to extend our work to other verticals to facilitate further research studying model generalization and domain adaptation.

[10] Some sites had less than 5 errors, in which case we examined all errors.

Acknowledgements

We would like to thank the anonymous reviewers for their detailed and insightful comments and suggestions. We would also like to thank Wendy Wei, Markus Dreyer, Polly Allen, Yusuke Watanabe, Matthias Petri, Tian Tang, Phi Nguyen, Ritesh Sarkhel, Hengxin Fun, Mengwen Liu, and Duke Vijitbenjaronk for their suggestions and assistance with this work.

Limitations

To ensure high quality extractions for PLAtE, we optimize our annotation process for precision. For example, for the Product Link attribute, we generally annotate only one product link per product. In an application scenario, the user would not need multiple links to a purchase page, but this could potentially harm the precision of the evaluated models. In addition, we assume that all attributes are text-based. This has the potential of missing additional product information which could be helpful to users, such as images of the product. In future work, we would like to extend PLAtE by incorporating other modalities.

References

Mohd Azir and Kamsuriah Ahmad. 2017. Wrapper approaches for web data extraction : A review. *2017 6th International Conference on Electrical Engineering and Informatics (ICEEI)*, pages 1–6.

Lidong Bing, Rui Guo, Wai Lam, Zheng-Yu Niu, and Haifeng Wang. 2014. Web page segmentation with structured prediction and its application in web page classification. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '14, page 767–776, New York, NY, USA. Association for Computing Machinery.

Lidong Bing, Wai Lam, and Tak-Lam Wong. 2013. Robust detection of semi-structured web records using a dom structure-knowledge-driven model. *ACM Trans. Web*, 7(4).

Michael J. Cafarella, Alon Y. Halevy, Hongrae Lee, Jayant Madhavan, Cong Yu, Daisy Zhe Wang, and Eugene Wu. 2018. Ten years of webtables. *Proc. VLDB Endow.*, 11:2140–2149.

Andrew Carlson and Charles Schafer. 2008. Bootstrapping information extraction from semi-structured web pages. In *Proceedings of the 2008th European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part I*, ECMLPKDD'08, page 195–210, Berlin, Heidelberg. Springer-Verlag.

Xiang Deng, Prashant Shiralkar, Colin Lockard, Binxuan Huang, and Huan Sun. 2022. Dom-lm: Learning generalizable representations for html documents. *arXiv preprint arXiv:2201.10608*.

Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2020. Turl: Table understanding through representation learning.

Paramveer S. Dhillon, Sundararajan Sellamanickam, and Sathiya Keerthi Selvaraj. 2011. Semi-supervised multi-task learning of structured prediction models for web information extraction. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, page 957–966, New York, NY, USA. Association for Computing Machinery.

Joseph L. Fleiss and Jacob Cohen. 1973. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and Psychological Measurement*, 33:613 – 619.

Tim Furche, Georg Gottlob, Giovanni Grasso, Xiaonan Guo, Giorgio Orsi, Christian Schallhart, and Cheng Wang. 2014. Diadem: Thousands of websites to a single database. *Proc. VLDB Endow.*, 7(14):1845–1856.

Tim Furche, Georg Gottlob, Giovanni Grasso, Giorgio Orsi, Christian Schallhart, and Cheng Wang. 2012. Amber: Automatic supervision for multi-attribute extraction.

Pankaj Gulhane, Amit Madaan, Rupesh Mehta, Jeyashankher Ramamirtham, Rajeev Rastogi, Sandeep Satpal, Srinivasan H Sengamedu, Ashwin Tengli, and Charu Tiwari. 2011. Web-scale information extraction with vertex. In *2011 IEEE 27th International Conference on Data Engineering*, pages 1209–1220.

Mansi Gupta, Nitish Kulkarni, Raghuveer Chanda, Anirudha Rayasam, and Zachary C Lipton. 2019. Amazonqa: A review-based question answering task. *arXiv preprint arXiv:1908.04364*.

Qiang Hao, Rui Cai, Yanwei Pang, and Lei Zhang. 2011. From one tree to a forest: A unified solution for structured web data extraction. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, page 775–784, New York, NY, USA. Association for Computing Machinery.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online. Association for Computational Linguistics.

Alexandra Hotti, Riccardo Sven Risuleo, Stefan Magureanu, Aref Moradi, and Jens Lagergren. 2021. The klarna product page dataset: A realistic benchmark for web representation learning.

Lawrence J. Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of Classification*, 2:193–218.

Nicholas Kushmerick. 1997. *Wrapper induction for information extraction*. University of Washington.

Jurek Leonhardt, Avishek Anand, and Megha Khosla. 2020. Boilerplate removal using a neural sequence labeling model. *Companion Proceedings of the Web Conference 2020*.

Junlong Li, Yiheng Xu, Lei Cui, and Furu Wei. 2021. Markuplm: Pre-training of text and markup language for visually-rich document understanding.

Bill Yuchen Lin, Ying Sheng, Nguyen Vo, and Sandeep Tata. 2020. Freedom: A transferable neural architecture for structured information extraction on web documents. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

G. Linden, B. Smith, and J. York. 2003. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Colin Lockard, Prashant Shiralkar, Xin Luna Dong, and Hannaneh Hajishirzi. 2020. Zeroshotceres: Zero-shot relation extraction from semi-structured webpages.

Petar Petrovski, Anna Primpeli, Robert Meusel, and Christian Bizer. 2016. The wdc gold standards for product feature extraction and product matching. In *EC-Web*, volume 278 of *Lecture Notes in Business Information Processing*, pages 73–86.

Victor Le Pochat, Tom van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. 2019. Tranco: A research-oriented top sites ranking hardened against manipulation. *Proceedings 2019 Network and Distributed System Security Symposium*.

Alexander Strehl and Joydeep Ghosh. 2002. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3:583–617.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Daheng Wang, Prashant Shiralkar, Colin Lockard, Binxuan Huang, Xin Luna Dong, and Meng Jiang. 2021. Tcn: Table convolutional network for web table interpretation. *Proceedings of the Web Conference 2021*.

Chenhao Xie, Wenhao Huang, Jiaqing Liang, Chengsong Huang, and Yanghua Xiao. 2021. Webke: Knowledge extraction from semi-structured web with pre-trained markup language model. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, page 2211–2220, New York, NY, USA. Association for Computing Machinery.

Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W. Bruce Croft. 2018. Towards conversational search and recommendation: System ask, user respond. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 177–186, New York, NY, USA. Association for Computing Machinery.

Shuyi Zheng, Ruihua Song, Ji-Rong Wen, and Di Wu. 2007. Joint optimization of wrapper generation and template detection. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, page 894–902, New York, NY, USA. Association for Computing Machinery.

Yichao Zhou, Ying Sheng, Nguyen Vo, Nick Edmonds, and Sandeep Tata. 2021. Simplified dom trees for transferable attribute extraction from the web.

Jun Zhu, Zaiqing Nie, Ji-Rong Wen, Bo Zhang, and Wei-Ying Ma. 2006. Simultaneous record detection and attribute labeling in web data extraction. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, page 494–503, New York, NY, USA. Association for Computing Machinery.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

Manuel Álvarez, Alberto Pan, Juan Raposo, Fernando Bellas, and Fidel Cacheda. 2008. Extracting lists of data records from semi-structured web pages. *Data & Knowledge Engineering*, 64(2):491–509.

A Training Details

We train all models for up to 5 epochs using an AdamW optimizer with a linear warm-up rate of 10% and decide on the best learning rate based on development set performance. We search over learning rates of: {5e-6, 5e-5, 5e-4}. The best learning rate was 5e-5 for both tasks for RoBERTa and attribute extraction for DOM-LM. 5e-6 was the best learning rate for both tasks for MarkupLM and segmentation for DOM-LM.

B Additional Tables

Attribute	Argmax Correct (%)
Product Name	89.4
Review Text	91.8
Product Link	16.2

Table 9: Percentage of false negative examples where the model would be correct if the $argmax$ of the attributes was chosen instead of requiring the activation to be above the threshold 0.5.

C Additional Figures

Godinger Lumina Hobnail DOF Glasses, Set of 4, $25

You guys, don't sleep on the selection of glassware at Bed, Bath & Beyond—you will be *shocked* at how many super-stylish and wildly affordable options they have. This hobnail set looks much more expensive than it is thanks to the cool, contemporary texture and shape of the glass. Be warned, your guests might think you've suddenly "come into some money" or something.

Figure 4: An example extraction "*Godinger Luminal Hobnail Glasses*" which is both a product name and product link.

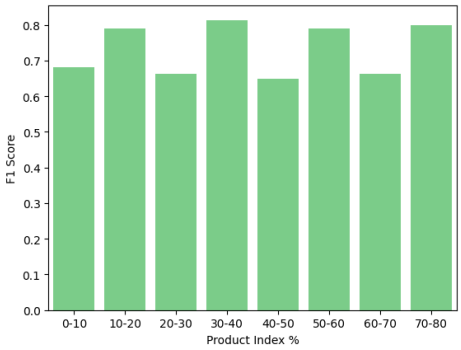

Figure 5: The average attribute extraction F1-score based on product index normalized by number of products for the MarkupLM model. The performance is relatively uniform across all indices indicating that product index does not have a significant effect on extraction performance. (We observed a similar trend for both RoBERTa and DOM-LM.)

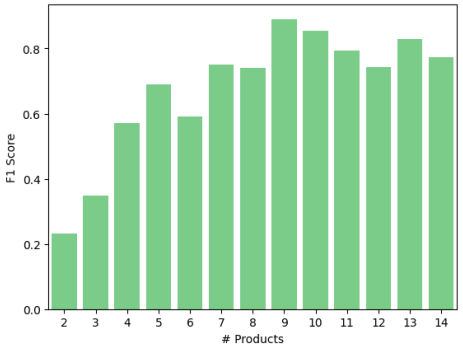

Figure 6: Average attribute extraction F1-score based on number of products in a page. In MarkupLM, as the number of products increases, F1-score also increases. (We observed a similar trend for both RoBERTa and DOM-LM.)

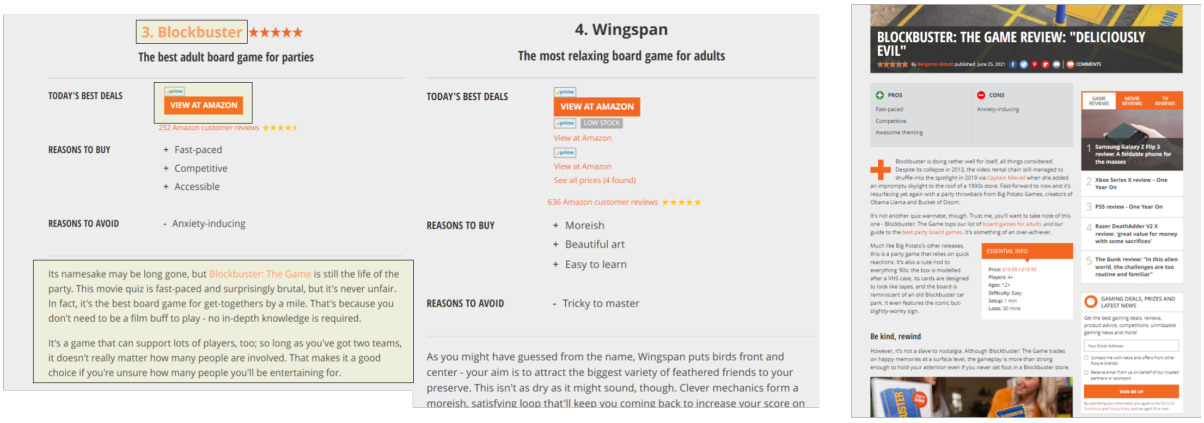

Figure 7: An example of a list page compared to a detail page. On the left is a list page with two products: “Blockbuster” and “Wingspan”. Both products have a similar format and directly comparable attributes from our schema: Product Name, Product Link, and Review Text. On the right is a detail page with more in-depth details and longer review text for the “Blockbuster” boardgame. (Screenshots from `gamesradar.com`)