PubSE: A Hierarchical Model for Publication Extraction from Academic Homepages

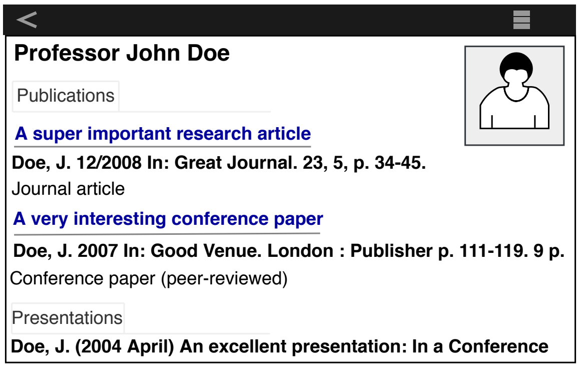
Yiqing Zhang **Jianzhong Qi** **Rui Zhang**[*] **Chuandong Yin**
School of Computing and Information Systems
The University of Melbourne
{yiqingz2,chuandongy}@student.unimelb.edu.au
{jianzhong.qi,rui.zhang}@unimelb.edu.au

Abstract

Publication information in a researcher's academic homepage provides insights about the researcher's expertise, research interests, and collaboration networks. We aim to extract all the publication strings from a given academic homepage. This is a challenging task because the publication strings in different academic homepages may be located at different positions with different structures. To capture the positional and structural diversity, we propose an end-to-end hierarchical model named PubSE based on Bi-LSTM-CRF. We further propose an alternating training method for training the model. Experiments on real data show that PubSE outperforms the state-of-the-art models by up to 11.8% in F1-score.

1 Introduction

Researchers often list their publications in their academic homepages. These publications provide insights about the researchers' expertise, research interests, and collaboration networks. Extracting publications from a researcher's homepage is an essential step in extracting the researcher's profile (Tang et al., 2010). In this study, we aim to extract every publication from a researcher's homepage. For ease of discussion, we call such a publication item a *publication string*. Figure 1 illustrates the studied problem. There are two publications on the homepage shown in the figure, a journal article and a conference paper. We aim to extract them as two separate publication strings.



Professor John Doe

Publications

A super important research article

Doe, J. 12/2008 In: Great Journal. 23, 5, p. 34-45.

Journal article

A very interesting conference paper

Doe, J. 2007 In: Good Venue. London : Publisher p. 111-119. 9 p.

Conference paper (peer-reviewed)

Presentations

Doe, J. (2004 April) An excellent presentation: In a Conference

Figure 1: Part of a homepage with publication strings

Extracting publication strings from academic homepages helps bypass the problem of name ambiguity (*i.e.*, different authors with the same name) (Zhu et al., 2018) in extracting publication strings from indexing sites such as DBLP and PubMed. However, extracting publication strings from academic homepages directly has its own challenges: (i) The list of publications may be located anywhere in a homepage with varying contexts. The structure of the list and the formatting styles of a publication string can vary vastly across different homepages. For example, some researchers like to list some of their publications with more details, such as full venue names, volume and page information, while listing the other publication in a concise way. Also, some researchers like to group their publication by year or by topic. (ii) A publication string may contain multiple lines of text (cf. Figure 1), and there may not be a clear boundary between two publications strings. (iii) There may be strings in an academic homepage that share very similar structures and styles with publication strings, such as records of conference presentations (cf. Figure 1). Previous work (Hong et al., 2009; Chung et al., 2012) focuses on feature and rule engineering and cannot accommodate the above challenges.

To address these challenges, we propose a model named *PubSE* to extract every publication string from an academic homepage. PubSE has two characteristics: (i) The model structure reflects the structure of a list of publications, by its loss-functions at both text line-level and webpage-level. (ii) The training process of the

[*] Corresponding author

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 1005–1010
Brussels, Belgium, October 31 - November 4, 2018. ©2018 Association for Computational Linguistics

model utilizes both text line-level and webpage-level information via an alternating training procedure to reduce overfitting. Our PubSE model can extract publication strings in non-trivial cases, such as multi-line publication strings from a non-continuous publication list. We make the following contributions: (i) We create a dataset of 2,500 homepages[1], in which each publication string is labeled. (ii) We address the problem of publication string extraction by end-to-end learning, without feature engineering. (iii) We propose a model that can learn the structures of publication lists and the styles of publication strings. We also propose an alternating training method that can reduce overfitting and further improve the prediction accuracy.

2 Related Work

Earlier studies extract publication strings from research papers (Peng and McCallum, 2006; Councill et al., 2008; Tkaczyk et al., 2015). Such a problem is simpler for two reasons: (i) The reference list of a research paper usually appears at a fixed position (e.g., end of paper) and is continuous. (ii) The references are usually well-formatted and have few format variations since they may be generated by software such as LaTeX.

For extracting publication strings from academic homepages, previous studies use either rule-based (Hong et al., 2009; Yang and Ho, 2010) or a hybrid of machine learning and rule-based methods (Chung et al., 2012).

For example, Chung et al. (2012) develop a system named PRM that first segments an HTML DOM tree and its contents based on HTML tags. Then, PRM uses a linear chain CRF model to label the different parts of the tree. Based on the labels, it refines the publication string boundaries by heuristic rules to produce final predictions.

Relying on the HTML DOM tree structure makes it difficult to train a machine learning based model for publication extraction because: (i) Text in a publication string may be separated in many different DOM tree nodes. (ii) The DOM tree structure, which previous web data record extraction systems (Liu et al., 2003; Furche et al., 2014; Omari et al., 2016) rely on, may vary given the same webpage content.

As a result, we do not use HTML tags in our model. Instead, we work on the visible text contents. To the best of our knowledge, no existing studies for publication string extraction can effectively model the structure of publication lists.

3 Dataset

To the best of our knowledge, there is no public dataset of academic homepages that has labeled publication strings.

We downloaded 2,500 academic homepages from 100 universities around the world. We use Selenium, an open-source automated rendering software, to render the webpages. We collect visible texts from the webpages and then manually tag all the publication strings in them. During tagging, we mark the beginning and ending byte offsets of each publication string. Among the 2,500 academic homepages, 723 homepages (28.9%) contain publication lists, which consist of a total of 13,237 publication strings. Among the 723 homepages that contain publication strings, there are 117 homepages (16.2%) that contain multi-line publication strings. On average, there are 732.1 (std=1583.3) tokens, 89.9 (std=141.6) lines, and 18.3 (std=35.4) publication strings per homepage. We call this dataset *HomePub*.

Each publication string in HomePub dataset is annotated by two annotators. Disagreement is resolved by a third annotator. On publication string level, annotators agree on 83.76% publications, and the Cohen's kappa is 0.2084.

We have developed a program *PageTagger*[2] to assist the annotation. On average, it takes about 2.5 minutes to annotate one academic homepage when using the PageTagger tool.

Note that our annotation does not consider the following as publication strings: (i) Master or PhD theses; (ii) working papers; (iii) seminars, invited talks, or presentations; and (iv) patents. Our annotation also excludes the numbers (e.g., [1] or [i]) if the publication strings are in a numbered list.

4 Methods

We summarize the baseline models in Section 4.1 and present our PubSE model in Section 4.2.

4.1 Baselines

PRM: Chung et al. (2012) develop a publication string extraction system based on region boundary

[1] Dataset available at `http://www.ruizhang.info/data/homepub.html`

[2] `https://github.com/yiqingzhang/page-tagger`

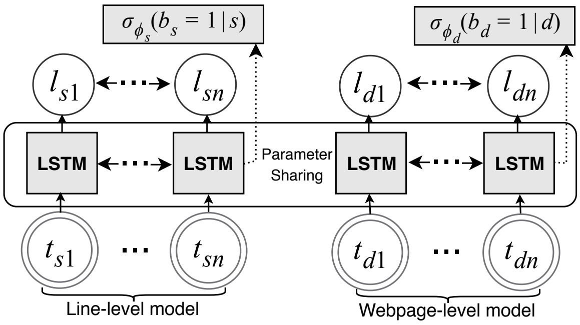


Figure 2: PubSE model

analysis and CRF. We train their system on 60% of the HomePub dataset.

ParsCit: ParsCit (Councill et al., 2008) is a CRF-based document structure analysis and reference parsing tool. We use ParsCit as an off-the-shelf tool on the HomePub dataset.

CNN-Sentence: In the HomePub dataset, over 80% of the publication strings are in a single line. The problem of extracting publication strings can be viewed as a single-line text classification problem (Kim, 2014; Yang et al., 2016; Joulin et al., 2017). Following Kim (2014), we implement a line-level classification model. We use the GloVe (300 dimensions) pre-trained embedding on this model (the same embedding is used across all the models that require word embeddings as the input). This model predicts whether each line in the webpage is a publication string.

Bi-LSTM-CRF: The problem of extracting publication strings can also be viewed as a sequence labeling task (Lample et al., 2016; Gui et al., 2017; Liu et al., 2017), where there are two possible labels for each token, publication (`I`) or non-publication (`O`). A consecutive sequence of `I` tokens forms a publications string. Sequence labeling approaches can capture correlations of labels and words, as well as words themselves. Following Ma and Hovy (2016), we implement a Bi-LSTM-CRF model. We concatenate token-level with character-level embeddings as the model input, to better deal with out-of-vocabulary tokens.

4.2 PubSE Model

To capture the hierarchical structure of a homepage and address the issue of multi-line publication strings, we propose a hierarchical model named *PubSE* with an alternating training method, which alternates between training a line-level and a webpage-level model to reduce overfitting. Figure 2 illustrates the model structure.

We incorporate both line-level and webpage-level information in the model since the predictions depend on both local (line-level) and global (webpage-level) context. On one hand, local information such as word embeddings and morphology information is crucial for the predictions over individual lines. On the other hand, global information is also necessary, e.g., to help identify the starting and ending positions of a publication list and the boundaries between multi-line publications strings.

Line-level model: As shown in Figure 2, the left Bi-LSTM network $\pi_\theta(\ell|s)$ specializes in line-level inputs, where each mini batch of input composes of lines in a webpage. On top of this sub-module, we add another layer $\sigma_{\phi_s}(b_s|s)$ to model whether each line contains a publication string:

$$\mathcal{L}_{line} = \lambda_{\theta_s}\mathcal{L}(\pi_\theta(\ell|s), y_t) + \lambda_{\phi_s}\mathcal{L}(\sigma_{\phi_s}(b_s|s), y_l),$$

where ℓ denotes the predicted labels for tokens t_{si} in a line s, and b_s denotes the predicted label for a line, *i.e.*, whether the line contains a publication string or not; y_t and y_l denote the ground truth label for each token and line. Hyperparameters λ_{θ_s} and λ_{ϕ_s} are the coefficients for token-level and line-level, while θ and ϕ_s are the parameters of the two networks; $\mathcal{L}$ is the cross-entropy loss:

$$\mathcal{L}(\hat{y}, y) = -(1-y)log(1-\hat{y}) - ylog(\hat{y}),$$

where $\hat{y}$ denotes the predicted label, and y denotes the ground truth label.

The inputs to the line-level model are each line in the form of $\langle(e_1, c_1), (e_2, c_2), ..., (e_n, c_n)\rangle$, where e_k and c_k are the word embedding and character embedding of the token t_{sk}. Network $\pi_\theta(\ell|s)$ outputs a label for each token, while network $\sigma_{\phi_s}(b_s|s)$ gives binary output for each line, indicating whether it contains a publication string. Extraction result is based on the output of $\pi_\theta(\ell|s)$.

Webpage-level model: Similarly, the right Bi-LSTM sub-module $\pi_\theta(\ell|d)$ in Figure 2 specializes in webpage-level inputs, where the whole homepage d is supplied to the model as a long sequence of token embeddings. We add another layer $\sigma_{\phi_d}(b_d|d)$ to reflect whether the homepage contains publication strings:

$$\mathcal{L}_{page} = \lambda_{\theta_d}\mathcal{L}(\pi_\theta(\ell|d), y_t) + \lambda_{\phi_d}\mathcal{L}(\sigma_{\phi_d}(b_d|d), y_w),$$

where b_d denotes whether the document contains publication or not; λ_{θ_d} and λ_{ϕ_d} are token-level and webpage-level coefficients; y_w denotes the ground truth label for a webpage; θ and ϕ_d are network parameters. Note that the left and right sub-modules collaborate by sharing network weights θ.

Methods	Exact Match			85% Match			Multi-line Only (Exact)		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
PRM (Chung et al., 2012)	15.39	37.54	21.83	18.25	44.52	25.89	5.70	10.83	7.47
ParsCit (Councill et al., 2008)	70.34	18.22	28.94	72.46	18.76	29.81	12.94	1.22	2.22
CNN-Sentence (Kim, 2014)	73.39	76.69	75.00	76.57	80.01	78.25	16.03	18.01	16.96
Bi-LSTM-CRF (Ma and Hovy, 2016)	74.15	77.22	75.65	75.76	78.90	77.30	21.11	22.21	21.65
PubSE (Proposed model)	**84.12**	**91.12**	**87.48**	**87.28**	**94.53**	**90.76**	**70.70**	**76.80**	**73.62**

Table 1: Performance of different models on the HomePub dataset. The first and second groups show the performance on the whole test dataset. The third group shows the exact matching performance on the subset that contains multi-line publications.

Alternating training method: Inspired by the training procedure of curriculum learning (Bengio et al., 2009) and soft-landing (He et al., 2016), we adopt an alternating training procedure controlled by the following function:

$$f(k) = H(cos(k/T)),$$

where T controls the period of the function, and k is the number of epochs. The H denotes the Heaviside step function. In the k^{th} epoch, we will train only one of the submodules, given by

$$\mathcal{L} = f(k)\mathcal{L}_{line} + f(k+1)\mathcal{L}_{page}.$$

Our intuition is that the training of line-level and webpage-level networks can reinforce each other and reduce overfitting. If we only train the model on the line-level input, the model will lose all the long-term dependency information. For example, a string that describes a thesis resembles that of a conference paper. To filter such strings, we need to rely on indicators that may reside in a different line such as a heading "Dissertations supervised". On the other hand, if we only train the model on webpage-level input, the model may be dominated by the longest line on the homepage, such as biography information. Our alternating training procedure balances the two factors and can better model the hierarchical structure of a publication list.

The PubSE model can capture and exploit information on the webpage from four different perspectives: (i) character-level information such as word morphology; (ii) token-level information such as word context; (iii) line-level information such as whether a line is a publication string; and (iv) webpage-level information such as whether a webpage contains publication strings.

5 Experiments

Settings and Evaluation: We divide the HomePub dataset by a 60-40 split and train our model on 60% of the total data. We use 20% of the training set as a validation set for early stopping and hyperparameter tuning. The optimal hyperparameters are obtained with a standard grid search procedure on the validation set. We use 40 as the batch size for the line-level model and 1 for the webpage-level model. We set $\lambda_{\theta_s}, \lambda_{\phi_s}, \lambda_{\theta_d}, \lambda_{\phi_d}, T$ as 1, 0.05, 1, 0.3, $1/\pi$, respectively.

We use precision, recall, and F1-score to measure the performance, and we report both exact and relaxed matching performance. In exact matching, a publication string is considered to be correctly extracted only if it exactly matches a publication string in the ground truth. In relax matching, we allow mismatching 15% of publication strings. (*i.e.*, a publication string is considered correct if it contains at least 85% of the tokens of a publication string in the ground truth.) We also list the model performance on webpages in the test set that contain multi-line publication strings.

Results: We report the experiment results in Table 1. The result shows that the proposed PubSE model consistently outperforms all the baselines with a statistically significant margin, and the advantage is up to 11.8%. In particular, the use of the webpage-level sub-module helps PubSE to handle multi-line publication strings, which yields a significant performance gain.

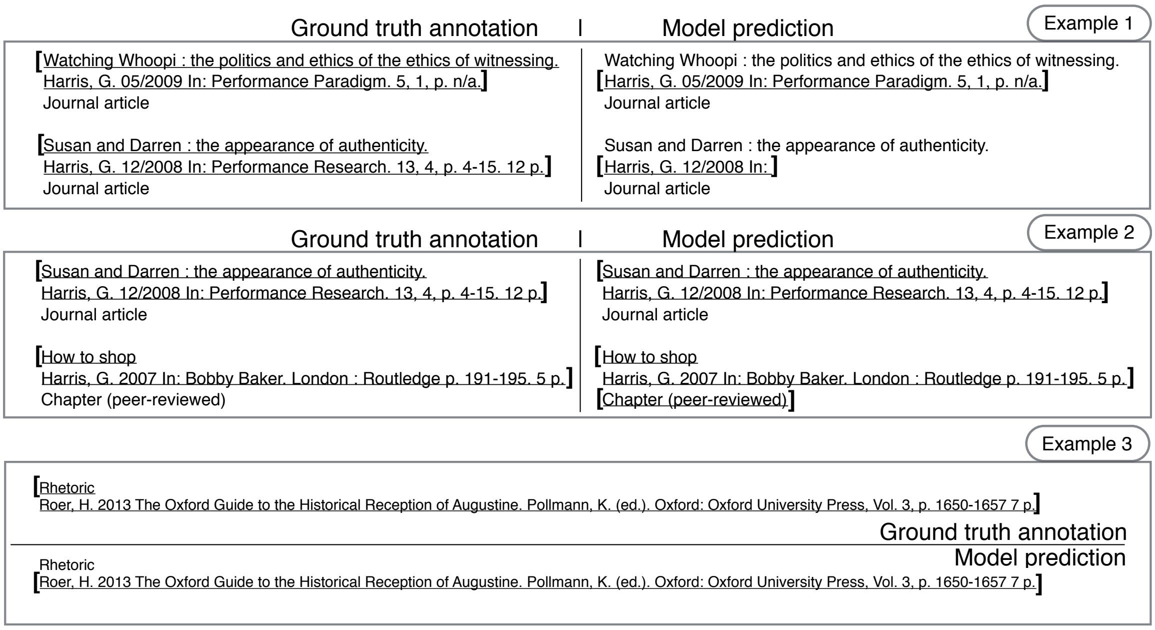
In comparison, PRM struggles in determining which part of a page contains publications. ParsCit requires well-formatted inputs. For example, if publication strings do not contain page numbers, ParsCit will be reluctant to separate the list of publication strings into individual records. CNN-Sentence and Bi-LSTM-CRF give poor results in pages that contain multi-line publication strings.

Methods	Exact Match	85% Match	Multi-line
L + LP	76.65	79.38	18.99
W	83.04	86.27	62.04
W + WP	84.18	87.22	63.77
L + W	85.98	89.1	63.36
PubSE (L+LP+W+WP)	87.48	90.76	73.62

Table 2: F1-score of different variations of the PubSE model. L means only the line-level sub-module $\pi_\theta(\ell|s)$; LP means the extra layer $\sigma_{\phi_s}(b_s|s)$; W means only the webpage-level sub-module $\pi_\theta(\ell|d)$; WP means the extra layer $\sigma_{\phi_d}(b_d|d)$.

Ablation Study: We also test different variations of our proposed model PubSE, and the results are shown in Table 2.

About 50% improvement over the best baseline



Example 1 — Ground truth annotation:

[Watching Whoopi : the politics and ethics of the ethics of witnessing.
Harris, G. 05/2009 In: Performance Paradigm. 5, 1, p. n/a.]
Journal article

[Susan and Darren : the appearance of authenticity.
Harris, G. 12/2008 In: Performance Research. 13, 4, p. 4-15. 12 p.]
Journal article

Example 1 — Model prediction:

Watching Whoopi : the politics and ethics of the ethics of witnessing.
[Harris, G. 05/2009 In: Performance Paradigm. 5, 1, p. n/a.]
Journal article

Susan and Darren : the appearance of authenticity.
[Harris, G. 12/2008 In:]
Journal article

Example 2 — Ground truth annotation:

[Susan and Darren : the appearance of authenticity.
Harris, G. 12/2008 In: Performance Research. 13, 4, p. 4-15. 12 p.]
Journal article

[How to shop
Harris, G. 2007 In: Bobby Baker. London : Routledge p. 191-195. 5 p.]
Chapter (peer-reviewed)

Example 2 — Model prediction:

[Susan and Darren : the appearance of authenticity.
Harris, G. 12/2008 In: Performance Research. 13, 4, p. 4-15. 12 p.]
Journal article

[How to shop
Harris, G. 2007 In: Bobby Baker. London : Routledge p. 191-195. 5 p.]
[Chapter (peer-reviewed)]

Example 3 — Ground truth annotation:

[Rhetoric
Roer, H. 2013 The Oxford Guide to the Historical Reception of Augustine. Pollmann, K. (ed.). Oxford: Oxford University Press, Vol. 3, p. 1650-1657 7 p.]

Example 3 — Model prediction:

Rhetoric
[Roer, H. 2013 The Oxford Guide to the Historical Reception of Augustine. Pollmann, K. (ed.). Oxford: Oxford University Press, Vol. 3, p. 1650-1657 7 p.]

Figure 3: Typical errors of various models. Underlined tokens are labeled as I (publication). “[” and “]” are not part of the text. They are used to highlight the boundary of publication strings.

is made by training the model with webpage-level input (W) since it is difficult to extract multi-line publication strings without a global view of the whole webpage.

The effect of the alternating training method (L+W) is also significant. The webpage-level model (W) may not handle short lines too well, e.g., a line with text “Conference paper (peer-reviewed)” as shown in Figure 1. This problem is solved by combining the line-level model with the webpage-level model (L+W).

Error analysis: Figure 3 shows typical errors made by various models. Examples 1 shows errors occurred in line-level model prediction results. The line-level model does not handle multi-line publication strings well since the predictions of different lines are independent, so the model fails to capture dependency relationships in different lines.

Example 2 shows prediction results given by the webpage-level model. We see that the webpage-level model can make a more accurate prediction for multi-line publications. However, it may make false positive predictions for short lines (e.g., “Chapter (peer-reviewed)”), while the line-level model seldom makes such mistakes. This is the motivation for us to integrate both the line-level and the webpage-level models.

PubSE can avoid most of the errors shown in Examples 1 and 2. Nevertheless, PubSE still makes mistakes in some challenging cases. Example 3 shows such a case, where PubSE does not recognize that “Rhetoric” is a publication title. A possible explanation is that such a short publication title is less common.

6 Conclusions and Future work

We studied publication string extraction and proposed a model named PubSE for the problem. PubSE models the publication list structure with its hierarchical structure and loss functions. We proposed an alternating training scheme that combines both line-level and webpage-level information, which are crucial for predicting multi-line publication strings. Experiments show that the proposed PubSE model outperforms the state-of-the-art models by up to 11.8% in F1-score.

For future work, we aim to expand our experimental study to a larger scale. We further consider extracting publication strings from academic homepages of the same organization. Such homepages may share similar templates, which may help improve the extraction accuracy. We also plan to investigate adaptive alternating model training schemes as well as external memory mechanism such as memory networks.

Acknowledgments

We thank Afshin Rahimi and the anonymous reviewers for their valuable suggestions, and we gratefully acknowledge the support of NVIDIA Corporation with the donation of the Tesla K40 GPU used for this research. We also sincerely thank Han Liu and other annotators for their help in creating the HomePub dataset.

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the Annual International Conference on Machine Learning (ICML)*, pages 41–48.

Jen-Ming Chung, Ya-Huei Lin, Hahn-Ming Lee, and Jan-Ming Ho. 2012. Mining publication records on personal publication web pages based on conditional random fields. In *Proceedings of the IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, pages 319–326.

Isaac G. Councill, C. Lee Giles, and Min-Yen Kan. 2008. ParsCit: an open-source CRF reference string parsing package. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, pages 661–667.

Tim Furche, Georg Gottlob, Giovanni Grasso, Xiaonan Guo, Giorgio Orsi, Christian Schallhart, and Cheng Wang. 2014. DIADEM: thousands of websites to a single database. *Proceedings of the VLDB Endowment (PVLDB)*, 7(14):1845–1856.

Tao Gui, Qi Zhang, Haoran Huang, Minlong Peng, and Xuanjing Huang. 2017. Part-of-speech tagging for twitter with adversarial neural networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2411–2420.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, pages 820–828.

Ching Hoi Andy Hong, Jesse Prabawa Gozali, and Min-Yen Kan. 2009. FireCite: Lightweight real-time reference string extraction from webpages. In *Proceedings of the Workshop on Text and Citation Analysis for Scholarly Digital Libraries (NLPIR4DL)*, pages 71–79.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 427–431.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 260–270.

Bing Liu, Robert Grossman, and Yanhong Zhai. 2003. Mining data records in web pages. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 601–606.

Fei Liu, Timothy Baldwin, and Trevor Cohn. 2017. Capturing long-range contextual dependencies with memory-enhanced conditional random fields. In *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, pages 555–565.

Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1064–1074.

Adi Omari, Benny Kimelfeld, Eran Yahav, and Sharon Shoham. 2016. Lossless separation of web pages into layout code and data. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1805–1814.

Fuchun Peng and Andrew McCallum. 2006. Information extraction from research papers using conditional random fields. *Information Processing and Management*, 42(4):963–979.

Jie Tang, Limin Yao, Duo Zhang, and Jing Zhang. 2010. A combination approach to web user profiling. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(1):1–38.

Dominika Tkaczyk, Paweł Szostek, Mateusz Fedoryszak, Piotr Jan Dendek, and Łukasz Bolikowski. 2015. CERMINE: automatic extraction of structured metadata from scientific literature. *International Journal on Document Analysis and Recognition (IJDAR)*, 18(4):317–335.

Kai-Hsiang Yang and Jan-Ming Ho. 2010. Parsing publication lists on the web. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, pages 444–447.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 1480–1489.

Jia Zhu, Xingcheng Wu, Xueqin Lin, Changqin Huang, Gabriel Pui Cheong Fung, and Yong Tang. 2018. A novel multiple layers name disambiguation framework for digital libraries using dynamic clustering. *Scientometrics*, 114(3):781–794.