Fast Few-shot Line-level Resume Dependency Parsing

Sean Liu and **Kevin Chang**
University of Illinois Urbana-Champaign
zxliu2@illinois.edu

Abstract

We propose a simple shift-based dependency parsing algorithm for academic resumes. The algorithm only uses positional data, and runs in linear time in relation to the document size. Training in a few-shot (around 10 resumes) setting yields strong performance (92% classification accuracy) despite using a low amount of information. We make the case that this algorithm is a strong baseline and serves as a proof-of-concept for further study.

1 Introduction

Document understanding has long been an active area of study, though until recently, the reading and understanding of documents have belonged to separate fields (computer vision; natural language programming). The former problem deals with the conversion of raw visual data into machine-readable formats such as raw text, JSON files, etc., while the latter tries to make sense of the semantics of the document itself, often requiring the document to be already processed. Now, multimodal models are emerging which combine the two tasks into an end-to-end framework, capable of reading in raw pixels (and perhaps a prompt) as input and performing complex tasks such as question answering on given that data (Huang et al., 2022; Lee et al., 2022b; Kim et al., 2021). Numerous datasets with varying parameters which all centre around this more advanced task of multimodal document understanding have also been developed (Mathew et al., 2021, 2022; Park et al., 2019).

We will be focusing on a specific subset of this problem: the task of *academic resume parsing*. We believe that the constraints of this problem are such that previous conventional methods are not sufficient to tackle it. In particular, we note the following about academic resumes:

1. Length: academics are often highly decorated and accomplished individuals, and their resumes reflect this fact: they often are upwards of ten pages long, with many resumes being around thirty pages. This presents an issue for transformers-based systems, as they often cannot handle extremely long documents due to memory constraints. While variations of transformers (Liu et al., 2021; Peng et al., 2021) have been developed that require less resources, common long-term dependencies makes a naive use of transformers a bad choice, since many optimisations are built on the assumption that long-distance dependencies are rare and do not affect performance.

2. Structure: As opposed to the diverse nature of documents that a general-purpose document understanding model may have to process, academic documents are generally computer-generated (as opposed to handwritten or scanned), single-columned, and well-behaved. Thus, more specialised heuristics may be employed to take advantage of these features.

In light of the above observations, we propose a transition-based dependency parser (Dozat and Manning, 2018) to extract the semantic relations between lines of a resume. We found that positional and stylistic data (bounding boxes, fonts) alone yields promising results, while using dense semantic representations from BERT (Devlin et al., 2018) confused the model, in a few-shot setting. Being able to extract the intrinsic tree *structure* of documents is an important stepping stone in document analysis, as the data may now be more readily analysed, for example, in extracting the education background of an individual, or listing out the publications of said individual. In addition, it may not always be feasible to employ large models, and in this context, smaller models approaching the performance of such models are of utmost importance in guaranteeing performance and efficiency. Recent work has shown blind scaling of models to be

suboptimal (Hoffmann et al., 2022); the pruning and distillation of models (Xia et al., 2022; Sanh et al., 2019) remain active areas of research.

In short, our main contributions can be summarised as follows:

1. The development and collection of a resume dataset for dependency analysis, along with a custom annotator;
2. showing that positional and stylistic information are strong predictors of structure within a resume;
3. extraction of not just the individual coherent elements of a document, but also the hierarchical relations between them;
4. showing that neural methods are effective in learning structural information even when the number of resumes is low (though each resume may have thousands of lines and transitions themselves).

2 Related Work

This work is built upon the foundation of much work that has been done in various other related fields, such as document segmentation, understanding, and dependency parsing.

2.1 Document Segmentation

Multiple segmentation algorithms have been proposed over the years, starting with heuristics such as geometric clustering and whitespace analysis - a recent survey is given in Binmakhashen and Mahmoud (2019). Cai et al. (2003) proposed a website segmentation algorithm which takes in visual and DOM Tree input. More recently, vision methods based on deep neural nets been dominant with architectures such as convolutional neural nets (CNNs, Li et al. (2021); Xu et al. (2021)) and vision transformers (ViTs, Han et al. (2022)) being the new state-of-the-art. More generally, this is a subset of the more general task of image segmentation, which aims to segment objects from all sorts of images (Kirillov et al., 2023). Note also that this method does not give the *relations* of the elements being extracted, just which elements there are.

2.2 Multimodal learning

Multimodal learning refers to the inputs to some algorithm not being restricted to one form of data - in our case, not just text or images. Radford et al. (2021) proposed to use contrastive learning to align text and image data, and recently, end-to-end document understanding systems such as LayoutLMv3 (Huang et al., 2022), Pix2Struct (Lee et al., 2022b), Formnet (Lee et al., 2022a), amongst others (Kim et al., 2022; Davis et al., 2023).

2.3 Dependency Parsing

Dependency parsing is one of the problems that used to compose the NLP pipeline but has mostly been phased out due to the power of large language models (LLMs), but saw a flurry of neural-powered development around a decade ago. Transition-based (Dozat and Manning, 2018) and graph-based (McDonald et al., 2005) algorithms are the two main paradigms to solve this task, and multiple augmentations to the two algorithms have been added to the base idea in the years following: StackLSTMs (Dyer et al., 2015), biaffine attention (Dozat and Manning, 2016), and more recently, stack transformers (Astudillo et al., 2020). Hwang et al. (2020) is closely related to, but ultimately differnet from, this work, parsing general documents using a graph-based approach.

3 Proposed Method

3.1 Problem Definition

We define a document as $\mathcal{D} = \{\ell_i\}_{i=1}^{N}$, where ℓ_i are the lines (or elements) of $\mathcal{D}$, and ℓ_0 is the root element, denoted ROOT. We will also assume that there is also an underlying *semantic tree* $\mathcal{T}(\mathcal{D})$ (denoted $\mathcal{T}$ when context is apparent). This tree is defined as a set of $M \leq N$ directed edges between the element: $\mathcal{T} = \{e_j = (u_j, v_j, t_j)\}_{j=1}^{M}$, where e_j is an edge from ℓ_{u_j} to ℓ_{v_j}, and t_j is one of two types:

1. t_j = SUBORDINATE: ℓ_{u_j} is a subordinate of ℓ_{v_j};
2. t_j = MERGE: ℓ_{u_j} belongs to the same semantic unit as ℓ_{v_j}, and their strings should be concatenated with ℓ_{v_j} first.

In addition, $\mathcal{T}$ should define a directed tree rooted at ℓ_0, but need not use all of the elements (for example, page numbers may be safely discarded without affecting the contents of the resume). Finally, we will assume that the order that the lines are in corresponds to some DFS ordering of $\mathcal{T}$. Our goal is to retrieve (or estimate) $\mathcal{T}$ given the document $\mathcal{D}$.

3.2 Method

3.2.1 Data Collection

As of the time of writing of this article, there is no currently known dataset of resumes with dependencies at the line level. As such, about 10 resumes were collected from University of Illinois professors in the Computer Science and Linguistics departments (each resume contained on the order of a few thousand lines, and so the total training instance size was quite large, if rather biased). Afterwards, a custom annotation and visualisation interface was written to enable manual annotation of these documents, and annotation of the resumes was completed.

3.2.2 Shift-based parsing algorithm

Following (Dozat and Manning, 2018), we implement a shift-based dependency parser with a stack and a buffer. The algorithm starts with the root element in the stack, and every iteration, the system decides one of the following actions to be taken:

1. POP: Pops the top element from the stack.
2. MERGE: Merges the current buffer element into the top stack element
3. SUBORDINATE: Indicates that the buffer element is a subordinate of the stack element; pushes the buffer element to the stack
4. DISCARD: Discards the current buffer element.

We may then formulate this problem as a *classification problem*: given some extracted features, can the system correctly classify the action to take? We measure performance using classification accuracy.

The algorithm terminates once the buffer is empty.

3.2.3 Feature engineering

We only used features from the current buffer element and the element at the top of the stack, and we extracted the following features:

1. Semantic information: for each element, we took the BERT (Devlin et al., 2018) embedding to encode semantic data. However, this turned out to be unfruitful, and we hypothesize that in addition to the data being scarce, text data from resumes are too highly formatted for BERT to extract useful information.
2. Positional information: we included positional information of the left, right, and height values of each box, the intuition being that the justification and font sizes are important in determining the relation of two elements. Following (Vaswani et al., 2017), we opted to pass in positional data not as raw numbers or percentages, but as a sinusodal vector of length 32.
3. Stylistic information: we also checked if the text was in bold or in italics, as we hypothesised that stylised text would correspond more to headers.

4 Experimental Results

Training with positional and stylistic information, we found that the model could achieve a training and validation classification error of 92%, while performance capped at around 70% if semantic information from BERT was included. We also tested on resumes that were not in the training set, and found that the results heavily depended on if the resumes were structured in a similar way as previously seen documents. We think that this is evidence in favour of the fact that structural information can be learnt effectively, and that the semantics of the document itself may not be that important in the extraction of structure; in addition this seems to imply that if more diverse resumes were introduced, then the model would be effective in extracting resume from a more diverse set of resumes in turn.

5 Conclusion

We have shown that typographical data can be a strong predictor of document structure, and that systems can be built which extract the relevant structure trees from said documents. This can be done in a few-shot environment without too much computational overhead - a 20-page long CV can be parsed in a few seconds on the author's laptop. It's important to note that this subtask can be seen as a generalisation of the usual segmentation task, as not only are the relevant sections extracted, but so are their hierarchical relations. Through the structure of a document, we may then run a number of different algorithms with greater effiency - for example, the quality and quantity of edges greatly affect graph neural network (GNN)-based networks (Scarselli et al., 2008; Gemelli et al., 2023), and a recent survey can be found in Wu et al. (2023).

6 Future Work

Although we have introduced a potential avenue of research, it is by no means conclusive, and many potential methods and applications remain to be investigated. For example, incorporating semantic or even raw visual data in the stack parsing algorithm, letting the algorithm see information from previous/posterior elements in both the stack and buffer (Dyer et al., 2015), and expanding to other domains are all potential research topics. Investigating additional applications of this general method may also prove fruitful.

References

Ramón Fernandez Astudillo, Miguel Ballesteros, Tahira Naseem, Austin Blodgett, and Radu Florian. 2020. Transition-based parsing with stack-transformers. *arXiv preprint arXiv:2010.10669*.

Galal M Binmakhashen and Sabri A Mahmoud. 2019. Document layout analysis: a comprehensive survey. *ACM Computing Surveys (CSUR)*, 52(6):1–36.

Deng Cai, Shipeng Yu, Ji-Rong Wen, and Wei-Ying Ma. 2003. Vips: a vision-based page segmentation algorithm.

Brian Davis, Bryan Morse, Brian Price, Chris Tensmeyer, Curtis Wigington, and Vlad Morariu. 2023. End-to-end document recognition and understanding with dessurt. In *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IV*, pages 280–296. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

Timothy Dozat and Christopher D Manning. 2018. Simpler but more accurate semantic dependency parsing. *arXiv preprint arXiv:1807.01396*.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A Smith. 2015. Transition-based parsing with stack long short-term memory. *arXiv preprint arXiv:1505.08075*.

Andrea Gemelli, Sanket Biswas, Enrico Civitelli, Josep Lladós, and Simone Marinai. 2023. Doc2graph: a task agnostic document understanding framework based on graph neural networks. In *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IV*, pages 329–344. Springer.

Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. 2022. A survey on vision transformer. *IEEE transactions on pattern analysis and machine intelligence*, 45(1):87–110.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. 2022. Layoutlmv3: Pre-training for document ai with unified text and image masking. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4083–4091.

Wonseok Hwang, Jinyeong Yim, Seunghyun Park, Sohee Yang, and Minjoon Seo. 2020. Spatial dependency parsing for semi-structured document information extraction. *arXiv preprint arXiv:2005.00642*.

Geewook Kim, Teakgyu Hong, Moonbin Yim, Jeongyeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. 2021. Ocr-free document understanding transformer. *arXiv preprint arXiv:2111.15664*.

Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. 2022. Ocr-free document understanding transformer. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVIII*, pages 498–517. Springer.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. *arXiv preprint arXiv:2304.02643*.

Chen-Yu Lee, Chun-Liang Li, Timothy Dozat, Vincent Perot, Guolong Su, Nan Hua, Joshua Ainslie, Renshen Wang, Yasuhisa Fujii, and Tomas Pfister. 2022a. Formnet: Structural encoding beyond sequential modeling in form document information extraction. *arXiv preprint arXiv:2203.08411*.

Kenton Lee, Mandar Joshi, Iulia Turc, Hexiang Hu, Fangyu Liu, Julian Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. 2022b. Pix2struct: Screenshot parsing as pretraining for visual language understanding. *arXiv preprint arXiv:2210.03347*.

Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. 2021. A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems*.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022.

Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. 2022. Infographicvqa. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1697–1706.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. 2021. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajic. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of human language technology conference and conference on empirical methods in natural language processing*, pages 523–530.

Seunghyun Park, Seung Shin, Bado Lee, Junyeop Lee, Jaeheung Surh, Minjoon Seo, and Hwalsuk Lee. 2019. Cord: a consolidated receipt dataset for post-ocr parsing. In *Workshop on Document Intelligence at NeurIPS 2019*.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah A Smith, and Lingpeng Kong. 2021. Random feature attention. *arXiv preprint arXiv:2103.02143*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2008. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Lingfei Wu, Yu Chen, Kai Shen, Xiaojie Guo, Hanning Gao, Shucheng Li, Jian Pei, Bo Long, et al. 2023. Graph neural networks for natural language processing: A survey. *Foundations and Trends® in Machine Learning*, 16(2):119–328.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. 2022. Structured pruning learns compact and accurate models. *arXiv preprint arXiv:2204.00408*.

Canhui Xu, Cao Shi, Hengyue Bi, Chuanqi Liu, Yongfeng Yuan, Haoyan Guo, and Yinong Chen. 2021. A page object detection method based on mask r-cnn. *IEEE Access*, 9:143448–143457.

A Appendix

A.1 The annotation interface

The annotation interface enables both human annotation (see 1) and the display of machine parsing results (see 2 and 3). The root element (in blue) is shown at the top of the first page, and lines are colour-coded based on tree depth. Green lines denote hierarchical relations (SUBORDINATE), and orange lines between nodes indicate a MERGE operation.

Comparing figures 2 and 3, we see that the former is much more like what we would expect the output to be, while the latter often gets confused. For example, it judges many lines in the Publications section to be their own elements, while they should be one coherent element; the second bad example shows the stack growing uncontrollably instead of being all siblings of a header node. In such cases, we hypothesise that it would benefit the model if additional information about the state of the stack or buffer were included.

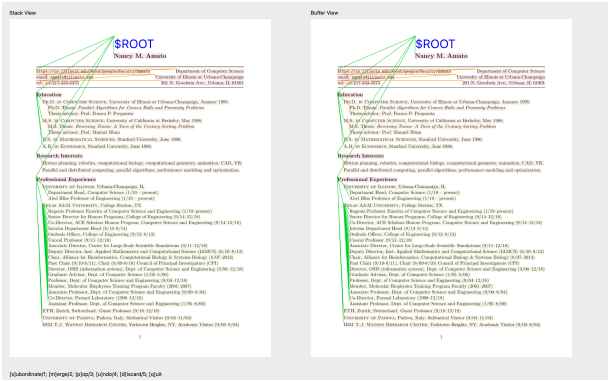

Figure 1: The annotation interface with a manual parsing result

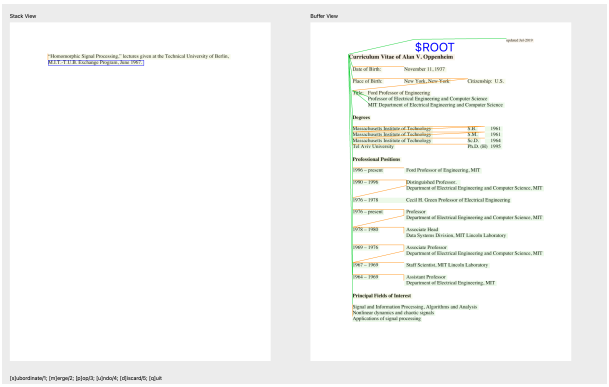

Figure 2: A better machine parsing result

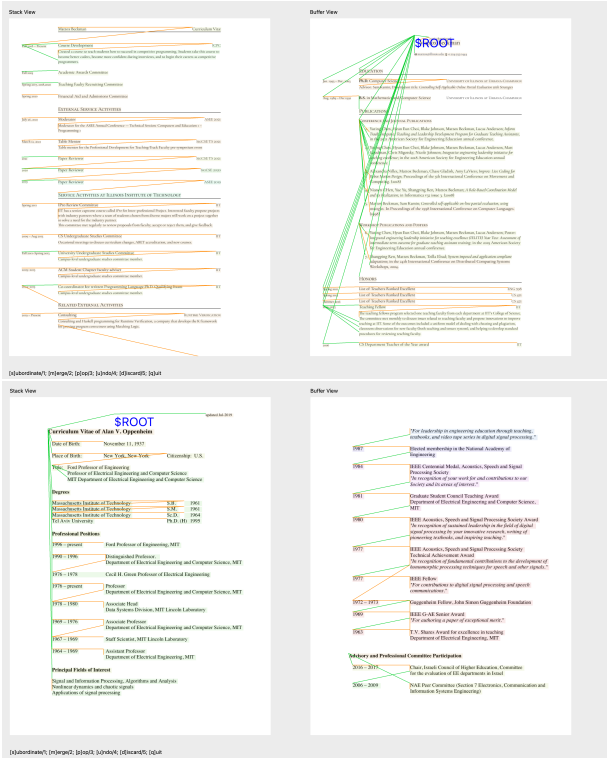

Figure 3: Worse machine parsing results