Deep Recurrent Neural Networks for Product Attribute Extraction in eCommerce

No Author Given

No Institute Given

Abstract. Extracting accurate attribute qualities from product titles is a vital component in delivering eCommerce customers with a rewarding online shopping experience via an enriched faceted search. We demonstrate the potential of Deep Recurrent Networks in this domain, primarily models such as Bidirectional LSTMs and Bidirectional LSTM-CRF with or without an *attention* mechanism. These have improved overall F_1 scores, as compared to the previous benchmarks [1] by at least 0.0391, showcasing an overall precision of 97.94%, recall of 94.12% and F_1 score of 0.9599. This has made us achieve a significant coverage of important facets or attributes of products which not only shows the efficacy of deep recurrent models over previous machine learning benchmarks but also greatly enhances the overall customer experience while shopping online.

Keywords: Sequence to Sequence Labeling, Bidirectional LSTM-CRF, Deep Recurrent Neural Networks, eCommerce Applications

1 Introduction

Problem Definition: Facets allow e-commerce customers to narrow down a search space, for example by restricting size of clothing or screen size of televisions. An exhaustive and accurate coverage of facets can ensure a pleasurable and efficient navigation experience through the given product space *iff* the attribute value metadata of the product properly appears in the facets for the given attribute.

Take S to be a search query entered by a user and $\mathcal{R}$ the set of products retrieved as result. Suppose product $\mathfrak{p} \in \mathcal{R}$, and $\mathfrak{p}$ has an attribute α which happens to be a facet. Suppose further that the value of α applicable to $\mathfrak{p}$ is v. When the user clicks on the attribute α with the associated facet value v, the filtered result set is $\mathcal{R}^{'} \subseteq \mathcal{R}$. Then $\mathfrak{p} \in \mathcal{R}^{'}$ if and only if $\mathfrak{p}(\alpha) = v$. An example of the problem statement is defined here.

Let x be a product title and let $(x_1, x_2, .., x_n)$ be a particular tokenization $\mathbf{x_t}$ of x. Given an attribute α, *attribute extraction* is the process of discovering the functions E_{seq} (raw extraction) such that

- $E_{seq}(\mathbf{x_t}) = E_{seq}((x_1, x_2, ..., x_n)) = (x_i, x_{i+1}, ..., x_k)$ for $1 \leq i \leq k \leq n$ where $\mathbf{a_v} = (x_i, x_{i+1}, ..., x_k)$ is a tokenization of a particular value of α

Example 1. Consider the product title:

$\mathbf{x_t}$ = Hewlett Packard B4L03A#B1H Officejet Pro Eaio

Let α be the attribute '***Brand***' we are interested in. Then we seek to find a function E (after whitespace tokenization) such that

$$E_{seq}((x_1, x_2, ..., x_6)) = (x_1, x_2) = (\text{Hewlett, Packard})$$

To achieve this solution, we implement a sequence to sequence labeling attribute extraction system which utilizes deep recurrent models at its core, including the Bidirectional LSTM Conditional Random Field with or without an attention mechanism. In traditional machine learning models, feature definition is an integral component to effective model performance [1]. However, deep models are able to learn the present features, without being predefined, ultimately allowing the system to be flexible in approximating sequence learning functions through new attributes. Examining the differences between the brands extracted by the two models, Table 1 shows a subset of the extraction discrepancies which are currently being targeted for corrections.

Table 1. Attribute Extraction Performance for '*Brand*' (Current vs. Previous Best [1])

Product Title	Previous Best	Current Deep Model
Woodland Imports Decorative Bottle	Woodland	**Woodland Imports**
Home Essentials White Essentials Sugar & Creamer	unbranded	**Home Essentials**
Plum Island Silver Sterling Silver Fairy Piece Ear Cuf	Plum Island	**Plum Island Silver**

Labeling Scheme for Annotating Product Titles: We describe our data annotation for training and validation purposes. In this example, we consider the attribute '*Brand*'. The BIO encoding scheme assigns one of the following three labels to each of the tokens in the title.

1. **B-attribute**: Token is the beginning token of an attribute value *iff* title contains the attribute value.
2. **I-attribute**: Token is the intermediate token of an attribute value, if exists.
3. **O**: Token is not representative of an attribute in the title.

To illustrate on a product title with the associated labels: 'The Green Pet Shop Self Cooling Dog Pad'.

The	Green	Pet	Shop	Self	Cooling	Dog	Pad
B-attribute	I-attribute	I-attribute	I-attribute	O	O	O	O

2 Model

Long Short Term Memory (LSTM) Network: Recurrent Neural Networks (RNN) are built to understand contextual significance, but fall short of this task

due to vanishing gradient problems wherein earlier parts of the network are less affected by backpropogation as compared to later parts of the network, resulting in convergence to suboptimal local minima [2]. Long Short Term Memory (LSTM) networks, which were adaptations of vanilla RNNs were introduced to address this problem by implementing a forget gate layer and a memory cell [3].

We use the following implementation, where σ is the logistic function, $\odot$ is an element-wise product and i, f, c, o and h are the *input gate*, *forget gate*, *cell*, *output gate* and *hidden* vectors respectively of same length. The weight matrix $\mathbf{W}$ has the corresponding subscripts for the different gates as the notation suggests. The LSTM cell generates a hidden vector $\mathbf{h}_t$ which at every time step abstracts the previous context.

$$
\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{W}_{ci}\mathbf{c}_{t-1} + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{W}_{ci}\mathbf{c}_{t-1} + \mathbf{b}_i) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot tanh(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{W}_{co}\mathbf{c}_{t-1} + \mathbf{b}_o) \\
\mathbf{h}_t &= \mathbf{o}_t \odot tanh(\mathbf{c}_t)
\end{aligned}
$$

Bidirectional LSTM-CRF: In previous work, notably Huang et. al. [4] proposed the usage of Bidirectional LSTM-CRF for named entity recognition by means of a sequence tagging task. Lample et. al. [5] also proposed a bidirectional LSTM with a sequential conditional random layer along with a new model of stack LSTMs with transition-based parsing. Similar models for sequence to sequence labeling models are frequently discussed in literature for various applications ([6], [7]).

The Bidirectional LSTM has garnered a lot of attention as it takes into context both past and future tokens when understanding the current token at time t [8]. Given the sequence of vectors $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n)$, the hidden vector serves as a concatenation of the hidden vectors from forward and backward states. If $\mathbf{h}_t^{left}$ denotes the hidden vector obtained from forward flowing states and $\mathbf{h}_t^{right}$ denotes the hidden vector obtained from backward flowing states, then the hidden representation of a token would be $\mathbf{h}_t^{B-LSTM} = [\mathbf{h}_t^{left}; \mathbf{h}_t^{right}]$.

The Conditional Random Field (CRF) is a probabilistic structured prediction model which predicts future labels, while taking into account previously predicted labels as well [9]. Lample et. al. [5] and Huang et. al. [4] both showed the advantages of CRF for label tagging using the features generated via the Bidirectional LSTM network as input parameters. An LSTM-CRF effectively uses LSTM layers to capture contextual information from the input sequence and a CRF at the output layer for efficient label tagging. This removes the need to hand engineer features for the CRF to learn. The CRF layer is learned by optimizing the parameters in the state transition matrix for the tags. Let's assume $\mathbf{M}$ is the matrix of scores given by the Bidirectional LSTM Network where $\mathbf{M}_i^j$ is the score of the j-th tag for the i-th token of the sequence. For a predicted sequence $(y_1, y_2, ..., y_n)$, where $\mathbf{A}$ is the transition matrix for the tags, the

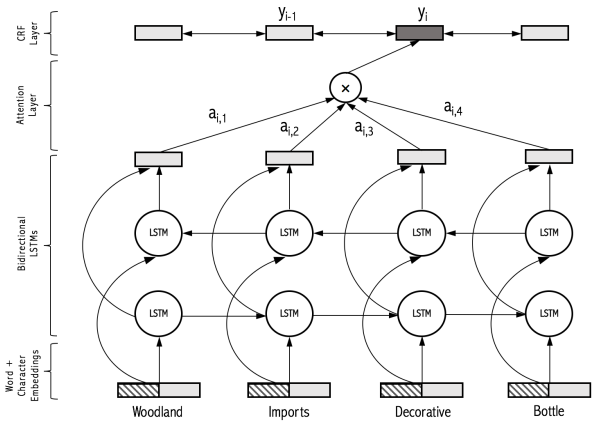

Fig. 1. Representation of Bidirectional LSTM-CRF Attention Model

combined score would be

$$s(\mathbf{x}, \mathbf{y}) = \sum_{i=0}^{n} \mathbf{A}_{y_i}^{y_{i+1}} + \sum_{i=1}^{n} \mathbf{M}_i^{y_i}$$

Finally, for all the methods described above, a softmax over all possible tags would provide the probabilities for the output tag sequence. The log-probability of the correct tag sequences is to be maximized during training [5]. Figure 1 captures the diagrammatic representations of the Bidirectional LSTM-CRF architecture.

Attention Mechanism: The attention mechanism [10] allows for an LSTM network to isolate tokens of contextual and locational interest from both past and future indices. Contextual information is used to understand tokens useful to the current index, and utilize those. Locational information complements this by allowing tokens to move around in memory, enabling the attention mechanism to persist through the entire network. As visualized in Figure 1, value for $a_{i,j}$ corresponds to respective attention weight prescribed by the j-th initial word token for output token y_i.

Word and Character Embeddings: A word embedding is a lower dimensional dense representation of a word which encodes not only the intrinsic meaning of the word but also the semantic meaning given its usage in various contexts. On the other hand, character embeddings [11] encapsulate patterns not explicitly noticeable through words. This for instance helps capture '*Brand*' attributes, in the case that the given brand does match with a respective word embedding. We consider both the word and character embeddings as random vectors for the initial run, but later allow the network to learn the embeddings based on the data and the task by means of an embedding layer.

3 Results

Dataset and Training Setup: We obtained product titles from online catalogs containing a variety of products. Our experiments pertaining to this paper concentrate around the attribute '*Brand*'. For '*Brand*', we collect 61,374 product titles for the experiment. Training, validation, and test data are generated with a 60/20/20 split ratio respectively. Titles are further tokenized by whitespace and labeled according to the annotation scheme described in Section 1. For accurate labels to train and validate our model, we acquire '*Brand*' attributes for the set of product titles through crowdsourcing tasks.

We employ stochastic gradient descent as a learning method to allow the gradient to back-propagate through time (BPTT). For all the deep models we consider word embeddings of size 100 and character embeddings of size 25. We add dropout layers with dropout rate 0.2 [12], and all models were run for 200 epochs with 5-fold cross validation. Higher dropout rate negatively affected our results.

Model Performance: Bidirectional LSTM with CRF layers outperforms all the other methods attempted as can be seen from Table 2. The F_1 measure for the Bidirectional LSTM-CRF is **0.9599** which is highest among all deep models. Compared to the previous best two models [1], Structured Perceptron and Linear Chain Conditional Random Field, the F_1 scores rose by **0.0392** and **0.0391** respectively. Even though attention did not positively impact the F_1 score for the Bidirectional LSTM-CRF, attention improved precision when applied over a Bidirectional LSTM without the CRF layer at the output.

Table 2. Model metrics for '*Brand*' Extraction with 5-fold Cross Validation

	Precision(%)	**Recall(%)**	**F_1-Score**	**Label Accuracy(%)**
Bidirectional-LSTM-CRF	**97.94**	**94.12**	**0.9599**	**99.44**
Bidirectional-LSTM-CRF Attention	97.38	93.98	0.9565	99.44
Bidirectional-LSTM Attention	95.12	92.80	0.9395	98.92
Bidirectional-LSTM	92.16	92.72	0.9244	98.92
Structured Perceptron [1]	91.98	92.18	0.9208	98.44
Linear Chain Conditional Random Field [1]	91.94	92.21	0.9207	98.44

Discussion: Using the attention mechanism does not guarantee higher precision for the Bidirectional LSTM-CRF. We speculate this being the case as the primary usage of a Bidirectional LSTM in this application is to deliver features to the CRF. The CRF is the final layer of the model which delivers an attribute label. The attention mechanism seems to interfere with the ability of the CRF layer to extract the attribute in question. However, in the absence of a CRF, with

a Bidirectional-LSTM model, attention helps the model performance, indicating that the attention mechanism and the CRF layer are both likely performing a similar function.

4 Conclusion

The attribute extraction system has significant impact on the product discoverability in a faceted search system which can be estimated from the online impressions on the products after a new set of attribute values have been added. In addition to '*Brand*' whose addition shows considerable improvement on clicks, add to cart rate and orders, other attributes also where attribute extraction was effective show a positive impact on impressions. '*Internal Memory*', '*Product Line*', and '*Manufacturer Part Number*' are examples of such attributes with 3.02%, 1.02% and 4.38% increases in precision scores respectively. Furthermore, to measure the impact, we conducted an experiment with a set of 272,697 products where approximately 250,000 additional impressions per day were observed over a period of 27 days after brand attribute values were added. This improvement in impressions is a result observed on the previous system. While our current method achieves 0.0391 increase in the F_1 score over the previous best methods, replacing previous models by the deep model potentiates a significant boost in impressions and other key metrics indicating notable business impacts.

References

1. More, A.: Attribute Extraction from Product Titles in eCommerce. EI-KDD (2016)
2. Bengio, Y., Simrad, P. and Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. on Neural Networks 5, 1013–1015 (1994)
3. Hochreiter, S., and Schmidhuber, J.: Long short-term memory. Neural Computation, (1997)
4. Huang, Z., Xu, W., and Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint, (2015)
5. Lample, G., Ballesteros, M., Subramanian, S. Kawakami, K. and Dyer, C.: Neural Architectures for Named Entity Recognition. Proc. of NAACL-HLT (2016)
6. Sutskever, I., Vinyals, O., V. Le, Quoc: Sequence to Sequence Learning with Neural Networks. NIPS (2014)
7. Dyer, C., Ballesteros, M., Ling, W., and Matthews, A., and Smith, N.A.: Transition based dependency parsing with stack long short-term memory. Proc. of ACL (2015)
8. Graves, A. and Schmidhuber, J.: Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures. Neural Networks (2005)
9. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. Proc. of ICML, (2001)
10. Olah, C, and Carter, S.: Attention and Augmented Recurrent Neural Networks. Distill, doi:10.23915/distill.00001, (2016)
11. Anand, A., Chakraborty, T., Park, N.: We used Neural Networks to Detect Clickbaits: You won't believe what happened next!. ECIR, Springer 541-547 (2017)
12. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Machine Learning (2014)