Spring 3-1-2020

Finding datasets in publications: the Syracuse University approach

Tong Zeng
Syracuse University

Daniel E. Acuna
Syracuse University

Recommended Citation

Zeng, Tong, & Acuna, Daniel E. (2020). Finding datasets in publications: the Syracuse University approach. In Rich Search and Discovery for Research Datasets (pp. 158–165). SAGE. http://doi.org/10.5281/zenodo.4402304

In J. Lane, I. Mulvany, P. Nathan (2020) Rich Search and
Discovery for Research Datasets (pp. 158-165) SAGE

11

FINDING DATASETS IN PUBLICATIONS: THE SYRACUSE UNIVERSITY APPROACH

Tong Zeng and Daniel Acuna

Introduction

Science is fundamentally an incremental discipline that depends on previous scientists' work. Datasets form an integral part of this process and therefore should be shared and cited like any other scientific output. This ideal is far from reality: the credit that datasets currently receive does not correspond to their actual usage (Zeng et al., 2020). One of the issues is that there is no standard for citing datasets, and even if they are cited, they are not properly tracked by major scientific indices. Interestingly, while datasets are still used and mentioned in articles, we lack methods to extract such mentions and properly reconstruct dataset citations. The Rich Context Competition challenge aims to close this gap by inviting scientists to produce automated dataset mention and linkage detection algorithms. In this chapter, we detail our proposal to solve the dataset mention step. Our approach attempts to provide a first approximation to better give credit and keep track of datasets and their usage.

The problem of dataset extraction has been explored before. Ghavimi et al. (2016, 2017) use a relatively simple TF-IDF representation with cosine similarity for matching dataset identification in social science articles. Their method consists of three major steps: preparing a curated dictionary of typical mention phrases, detecting dataset references, and ranking matching datasets based on cosine similarity of TF-IDF representations. This approach achieved a relatively high performance, with $F_1 = 0.84$ for mention detection and $F_1 = 0.83$, for matching. Singhal and Srivastava (2013) proposed a method using normalized Google distance to screen whether a term is in a dataset. However, this method relies on external services and is not computationally efficient. They achieve a good $F_1 = 0.85$ using Google search and $F_1 = 0.75$ using Bing. A somewhat similar project was proposed by Lu et al. (2012). They built a dataset search engine by solving the two challenges: identification of the dataset and association to a URL. They build a dataset of 1000 documents with their URLs, containing 8922 words or abbreviations representing datasets. They also build a web-based interface. This shows the importance of dataset mention extraction and how several groups have tried to tackle the problem.

In this chapter, we describe a method for extracting dataset mentions based on a deep recurrent neural network. In particular, we used a bidirectional long short-term memory (bi-LSTM) sequence to sequence model paired with a conditional random field (CRF) inference mechanism. The architecture is similar to that of Chapter 6, but we only focus on the detection of dataset mentions. We tested our model on a novel dataset produced for the Rich Context Competition challenge. We achieved a relatively good performance of $F_1 = 0.885$. We discuss the limitations of our model.

The Dataset

The rich context dataset challenge was proposed by the New York University's Coleridge Initiative.[1] The challenge comprised several phases, and participants moved through the phases depending on their performance. We only analyse data from the first phase. This phase contained a list of datasets and a labelled corpus of around 5000 publications. Each publication was labelled indicating whether a dataset was mentioned within it and which part of the text mentioned it. The challenge used an accuracy measure for measuring the performance of the competitors and also the quality of the code, documentation, and efficiency.

We adopted the CoNLL 2003 format (Tjong Kim Sang and De Meulder, 2003) to annotate whether a token is a part of dataset mention. Concretely, we use the tag DS to denote a dataset mention; a B- prefix indicates that the token is the beginning of a dataset mention, an I- prefix indicates that the token is inside a dataset mention, and O denotes a token that is not a part of a dataset mention. We put each token and its tag (separated by a horizontal tab control character) on one line, and use the end-of-line (\n) control character as separator between sentences (see Table 11.1). The dataset was randomly split by 70%, 15%, 15% for training set, validation set and testing set, respectively.

Table 11.1 Example of a sentence annotated by IOB tagging format

Token	Annotation
This	O
…	
data	O
from	O
the	O
Monitoring	B-DS
the	I-DS
Future	I-DS
(	O
MTF	B-DS
)	O
\n	

The Proposed Method

Overall View of the Architecture

In this section we propose a model for detecting mentions based on a bi-LSTM CRF architecture. At a high level, the model uses a sequence-to-sequence recurrent neural network that produces the probability of whether a token belongs to a dataset mention. The CRF layer takes those probabilities and estimates the most likely sequence based on constraints between label transitions (e.g., mention–to–no-mention–to–mention has low probability). While this is a standard architecture for modelling sequence labelling, the application to our particular dataset and problem is new.

We now describe in more detail the choices of word representation, hyperparameters and training parameters. A schematic view of the model is given in Figure 11.1 and the components are as follow.

1 Character encoder layer: treat a token as a sequence of characters and encode the characters by using a bi-LSTM to get a vector representation.
2 Word embedding layer: mapping each token into fixed-size vector representation by using a pre-trained word vector.
3 Bi-LSTM layer: make use of bi-LSTM network to capture the high-level representation of the whole token sequence input.
4 Dense layer: project the output of the previous layer to a low-dimensional vector representation of the distribution of labels.
5 CRF layer: find the most likely sequence of labels.

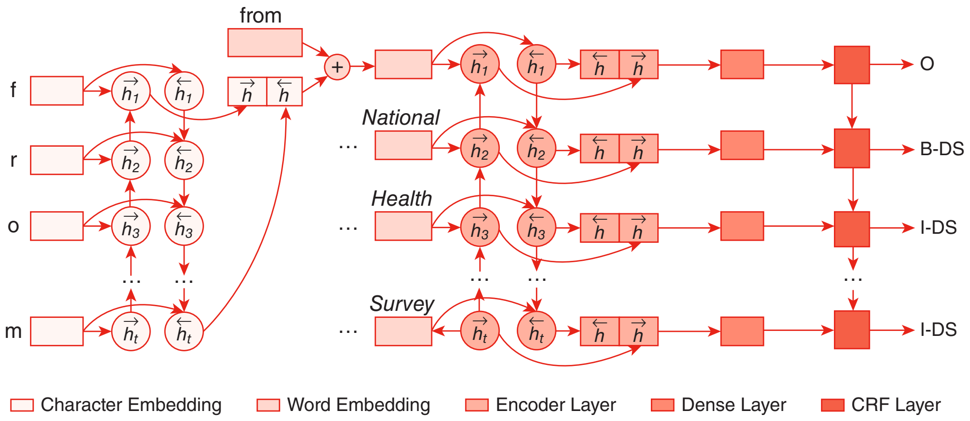

Figure 11.1 Network architecture of bi-LSTM CRF network

Character Encoder

Similar to the bag of words assumption, a word could be composed of characters sampled from a bag of characters. Previous research (Santos and Zadrozny, 2014; Jozefowicz et al., 2016) has shown that the use of character-level embedding could benefit multiple NLP-related tasks. In order to use character-level information, we break down a word into a sequence of characters, then build a vocabulary of characters. We initialize the character embedding weights using the vocabulary size of a pre-defined embedding dimension, then update the weights during the training process to get the fixed-size character embedding. Next, we feed a sequence of the character embedding into an encoder (a bi-LSTM network) to produce a vector representation of a word. By using a character encoder, we can solve the out-of-vocabulary problem for pre-trained word embedding, as every word could be composed of characters.

Word Embedding

The word embedding layer is responsible for storing and retrieving the vector representation of words. Specifically, the word embedding layer contains a word embedding matrix $M^{\text{tkn}} \in \mathbb{R}^{|V|d}$, where V is the vocabulary of the tokens and d is the size of the embedding vector. The embedding matrix was initialized by a pre-trained GloVe vectors (Pennington et al., 2014), and updated by learning from the data. In order to retrieve from the embedding matrix, we first convert a given sentence into a sequence of tokens, then for each token we look up the embedding matrix to get its vector representation. Finally, we get a sequence of vectors as input for the encoder layer.

LSTM

A recurrent neural network (RNN) is a type of artificial neural network which recurrently takes the output of the previous step as input of the current step. This recurrent nature allows it to learn from sequential data, for example, the text which consists of a sequence of works. An RNN could in theory capture contextual information in variable-length sequences, but it suffers from gradient exploding/vanishing problems (Pascanu et al., 2013). The long short-term memory (LSTM) architecture was proposed by Hochreiter and Schmidhuber (1997) to cope with these gradient problems. Similar to a standard RNN, the LSTM network also has a repeating module called an LSTM cell. The cell remembers information over arbitrary time-steps because it allows information to flow along it without change. The cell state is regulated by a forget gate and an input gate which control the proportion of information to forget from a previous time-step and to remember for a next time-step. Also, there is an output gate controlling the information flowing out of the cell. The LSTM could be defined formally by the following equations:

$$
\begin{aligned}
i_t &= \sigma(W_i x_t + W_i h_{t-1} + b_i),\\
f_t &= \sigma(W_f x_t + W_f h_{t-1} + b_f),\\
g_t &= \tanh(W_g x_t + W_g h_{t-1} + b_g),\\
o_t &= \sigma(W_o x_t + W_o h_{t-1} + b_o),\\
c_t &= f_t \otimes c_{t-1} + i_t \otimes g_t,\\
h_t &= o_t \otimes \tanh(c_t),
\end{aligned}
$$

where x_t is the input at time t, W is the weights, b is the bias. The σ is the sigmoid function, $\otimes$ denotes the dot product, c_t is the LSTM cell state at time t and h_t is hidden state at time t. i_t, f_t, o_t and g_t are referred to as input, forget, output and cell gates, respectively.

LSTM can learn from the previous steps, which is the left context if we feed the sequence from left to right. However, the information in the right context is also important for some tasks. The bi-LSTM (Graves et al., 2013) satisfies this information need by using two LSTMs. Specifically, one LSTM layer was fed by a forward sequence and the other by a backward sequence. The final hidden states of each LSTM were concatenated to model the left and right contexts:

$$h_t = \left[\overrightarrow{h_t}\ \overleftarrow{h_t}\right].$$

Finally, the outcomes of the states are taken by a CRF layer (Lafferty et al., 2001) that takes into account the transition nature of the beginning, intermediate, and end of mentions.

Results

In this work we wanted to propose a model for the Rich Context Competition challenge. We propose a relatively standard architecture based on the bi-LSTM CRF network. We now describe the evaluation metrics, hyperparameter setting, and the results of this network on the dataset provided by the competition.

For all of our results, we use F_1 as the measure of performance. This measure is the harmonic average of the precision and recall and it is the standard measure used in sequence labelling tasks. It varies from 0 to 1, the higher the better. Our method achieved a relatively high F_1 of 0.885 for detecting mentions.

Table 11.2 Model search space and best assignments

Hyperparameter	Search space	Best parameter
Number of epochs	50	50
Patience	10	10
Batch size	64	64
Pre-trained word vector size	choice[50, 100, 200,300]	100
Encoder hidden size	300	300
Number of encoder layers	2	2
Dropout rate	choice[0.0,0.5]	0.5
Learning rate optimizer	Adam	Adam
L2 regularizer	0.01	0.01
Learning rate	0.001	0.001

We train models using the training data and monitor the performance using the validation data (we stop training if the performance does not improve for the last 10 epochs). We use the Adam optimizer with learning rate 0.001 and batch size equal to 64. The hidden size of LSTM for character and word embedding is 80 and 300, respectively. For the regularization methods, and to avoid overfitting, we use L2 regularization set to 0.01 and we also use dropout rate equal to 0.5. We trained eight models with a combination of different GloVe vector size (50, 100, 300 and 300) and dropout rate (0.0, 0.5). The hyperparameter settings are shown in Table 11.2.

Table 11.3 Performance of proposed network

Model	GloVe size	Dropout rate	Precision	Recall	F_1
1	50	0.0	0.884	0.873	0.878
2	50	0.5	0.877	0.888	0.882
3	100	0.0	0.882	0.871	0.876
4	100	0.5	0.885	0.885	0.885
5	200	0.0	0.882	0.884	0.883
6	200	0.5	0.885	0.880	0.882
7	300	0.0	0.868	0.886	0.877
8	300	0.5	0.876	0.878	0.877

The test performances are reported in Table 11.3. The best model is trained by word vector size 100 and dropout rate 0.5, with F_1 score 0.885 (Table 11.3), and it takes 15 hours 58 minutes for the training on an NVIDIA GTX 1080 Ti GPU in a computer with an Intel Xeon E5-1650v4 3.6 GHz CPU with 128GB of RAM.

We also found some limitations to the dataset. Firstly, we found that mentions are nested (e.g. HRS, RAND HRS, RAND HRS DATA are linked to the same dataset). The second issue is that most of the mentions have ambiguous relationships to datasets. In particular, only 17,267 (16.99%) mentions are linked to one dataset, 15,292 (15.04%) mentions are linked to two datasets, and 12,624 (12.42%) are linked to three datasets. If these difficulties are not overcome, then the predictions from the linkage process will be noisy and therefore impossible to tell apart.

Conclusion

In this work, we report a high-accuracy model for the problem of detecting dataset mentions. Because our method is based on a standard bi-LSTM CRF architecture, we expect that updating our model with recent developments in neural networks would only benefit our results. We also provide some evidence of how difficult we believe the linkage step of the challenge could be if dataset noise is not lowered.

One of the shortcomings of our approach is that the architecture is lacking some modern features of RNN networks. In particular, recent work has shown that attention mechanisms are important especially when the task requires spatially distant information, as in this case. These benefits could also translate to better linkage. We are exploring new architectures using self-attention and multiple-head attention. We hope to share these approaches in the near future.

There are a number of improvements that we could make in the future. A first improvement would be to use non-recurrent neural architectures such as the Transformer which has been shown to be faster and a more effective learner than RNNs. Another improvement would be to bootstrap information from other dataset sources such as open-access full-text articles from PubMed Open Access Subset. This dataset contains dataset *citations* (Zeng et al., 2020) – in contrast to the most common types of citations to publications. The location of such citations within the full text could be exploited to perform entity recognition. While this would be a somewhat different problem than the one solved in this chapter, it would still be useful for the goal of tracking dataset usage. In sum, by improving the learning techniques and the dataset size and quality, we could significantly increase the success of finding datasets in publications.

Our proposal, however, is surprisingly effective. Because we have barely modified a general RNN architecture, we expect that our results will generalize relatively well either to the second phase of the challenge or even to other disciplines. We would emphasize, however, that the quality of the dataset has a great deal of room for improvement. Given how important this task is for the whole of science, we should strive to improve the quality of these datasets so that techniques like this one can be more broadly applied.

Acknowledgements

Tong Zeng was funded by the China Scholarship Council #201706190067. Daniel E. Acuna was funded by the National Science Foundation awards #1646763 and #1800956.

References

Ghavimi, B., Mayr, P., Lange, C., Vahdati, S. and Auer, S. (2017) A semi-automatic approach for detecting dataset references in social science texts. *Information Services & Use*, 36(3–4), 171–187.

Ghavimi, B., Mayr, P., Vahdati, S. and Lange, C. (2016) Identifying and improving dataset references in social sciences full texts. Preprint, arXiv:1603.01774 [cs].

Graves, A., Mohamed, A. and Hinton, G. (2013) Speech recognition with deep recurrent neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. Piscataway, NJL IEEE, pp. 6645--6649.

Hochreiter, S. and Schmidhuber, J. (1997) Long short-term memory. *Neural Computation*, 9(8), 1735–1780.

Jozefowicz, R., Vinyals, O., Schuster, M., Shazeer, N. and Wu, Y. (2016) Exploring the limits of language modeling. Preprint, arXiv:1602.02410.

Lafferty, J., McCallum, A. and Pereira, F. C. N. (2001) Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*. San Francisco: Morgan Kaufmann, pp. 282–289.

Lu, M., Bangalore, S., Cormode, G., Hadjieleftheriou, M. and Srivastava, D. (2012) A dataset search engine for the Research Document Corpus. In *2012 IEEE 28th International Conference on Data Engineering*. Piscataway, NJ: IEEE, pp. 1237–1240.

Pascanu, R., Mikolov, T. and Bengio, Y. (2013) On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, Stroudsburg, PA: International Machine Learning Society. pp. 1310–1318.

Pennington, J., Socher, R. and Manning, C. (2014) GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Red Hook, NY: Curran, pp. 1532–1543.

Santos, C. D. and Zadrozny, B. (2014) Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. Red Hook, NY: Curran, pp. 1818–1826.

Singhal, A. and Srivastava, J. (2013) Data Extract: Mining context from the Web for dataset extraction. *International Journal of Machine Learning and Computing*, 3(2), 219–223.

Tjong Kim Sang, E. F and De Meulder, F. (2003) Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, Volume 4. Morristown, NJ: Association for Computational Linguistics, pp. 142–147.

Zeng, T., Wu, L., Bratt, S. and Acuna, D. E. (2020) Assigning credit to scientific datasets using article citation networks. *Journal of Informetrics*, 14(2), 101013.

Notes

[1] https://coleridgeinitiative.org/richcontextcompetition