Encoder-Decoder neural networks for taxonomy classification

Makoto Hiramatsu
Graduate School of Library, Information and Media Studies, University of Tsukuba
Tsukuba, Ibaraki
himkt@klis.tsukuba.ac.jp

Kei Wakabayashi
Faculty of Library, Information and Media Science, University of Tsukuba
Tsukuba, Ibaraki
kwakaba@slis.tsukuba.ac.jp

ABSTRACT

This paper describes our taxonomy classifier for SIGIR eCom Rakuten Data Challenge. We propose a taxonomy classifier based on sequence-to-sequence neural networks, which are widely used in machine translation and automatic document summarization, by treating taxonomy classification as the translation problem from a description of a product to a category path. Experiments show that our method can predict category paths more accurately than baseline classifier.

CCS CONCEPTS

• **Computing methodologies → Information extraction**;

KEYWORDS

Encoder-Decoder Neural Networks, Recurrent Neural Networks, Taxonomy classification

ACM Reference Format:
Makoto Hiramatsu and Kei Wakabayashi. 2018. Encoder-Decoder neural networks for taxonomy classification. In *Proceedings of ACM SIGIR Workshop on eCommerce (SIGIR 2018 eCom Data Challenge).* ACM, New York, NY, USA, Article 4, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Taxonomy is the major classification schemes in organizing concepts. With the rapid growth of the e-commerce market accompanying the development on the Internet, the number of products on e-commerce becomes enormous. In this situation, it is required to develop methods that predict taxonomic categories automatically because it is costly to classify all the products manually.

Rakuten Data Challenge, which is a competition we participated, provides a task to predict correct categories for each given product. As a feature of this task, categories have a hierarchical structure. This hierarchical structure corresponds to a taxonomy, which indicates that items in a category are further classified into a subcategory that contains further lower detail information. Each product has a path in the taxonomy like "Clothing, Shoes & Accessories → Shoes → Men → Boots".

As an approach to solving this task, the most straightforward approach is to train a multi-class classifier (e.g., Random Forest) that predicts a category path as a class of a given product. However, as mentioned earlier, the number of category paths is 3,695, which is fairly large to be considered as a set of classes for ordinal machine learning classifier. Moreover, this approach independently treats these category paths although a category path shares a part of another category path of a similar product. It is expected that this fact causes more data sparseness issue and degrades the performance because the classifier has no way to find common patterns that are shared in two different category paths.

In this paper, we propose a taxonomy classifier based on Encoder-Decoder neural networks. The key idea is to regard the category path as a series of category names in each hierarchical level. From this perspective, the taxonomy classification task can be converted into a sequence-to-sequence problem, which has a text (i.e., a sequence of words) of the product name as the input and a sequence of category names as the output. In recent years, remarkable performance has been demonstrated in the field of machine translation and automatic summarization by using the model called neural network Encoder-Decoder architecture. We apply the Encoder-Decoder model to the taxonomy classification task and evaluate the performance. Experiments show that our approach can successfully predict category paths more precisely than the baseline approach that treats the task as a multi-class classification problem and applies Random Forest.

2 DATASET

Table 1: Histogram of the depth of category paths

Category depth	Frequency of item
1	8,172
2	2,792
3	228,888
4	344,472
5	166,165
6	45,253
7	4,197
8	61

We have 800,000 records for training data and 200,000 records for test data. Each record has a description of a product and a category path. The number of labels in the training data is 3,695, and each label is assigned to 868 items on average. The category (id=4015) is most frequently assigned to products, which is assigned to 268,295 items. Figure 1 shows the histogram of the number of words in each description in the training data. The average number of words was 10.92, and the standard deviation was 5.21. The maximum number of words was 58, and the minimum value was 1.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

SIGIR 2018 eCom Data Challenge, July 2018, Ann Arbor, Michigan, USA
© 2018 Copyright held by the owner/author(s).
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

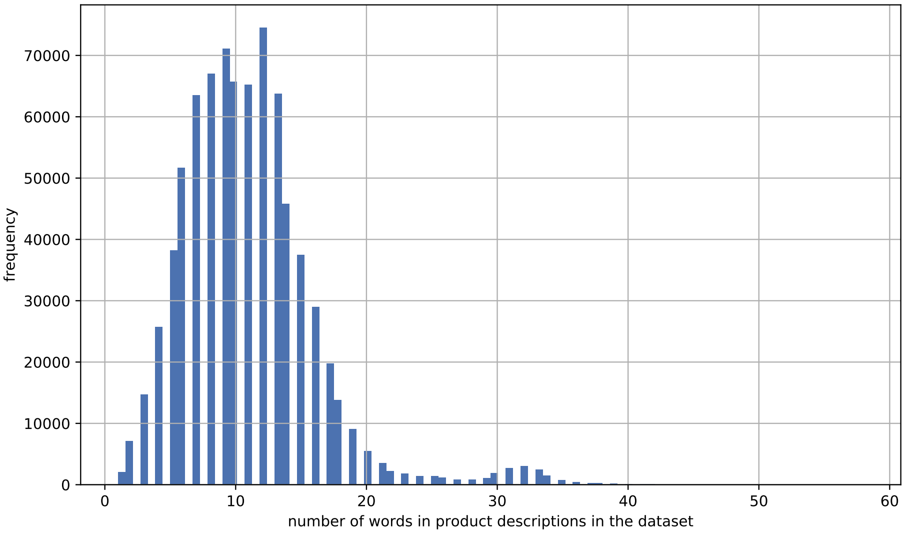

Figure 1: Histogram of the number of words in product descriptions in the dataset

Table 1 shows the histogram of the depth of category path in the training set. The depth of category path in training set is 4.01 on average. In other words, each product has four categories on average. The maximum depth of the depth of category path was 8, and the minimum depth was 1.

3 PROPOSED METHOD

3.1 Preprocessing

We used 20 % of the training dataset as the validation set to evaluate models. As preprocessing, we lowercase a product name in training/validation/test sets with SpaCy [1]. We use both the original corpus and the lowercase corpus and compare classifier performances.

For the weights of dense word representation layer, we use GloVe [6] pre-trained embeddings trained on Gigaword and Wikipedia. GloVe contains the lowercase words in its vocabulary. The preprocessing of lowercase makes the vocabulary *matching rate* improve. We show the *matching rate* of two corpora in Table 2 where source means descriptions of products, which are inputs. *matching rate* is defined by

$$matching\ rate = \frac{|V_{Dataset} \cap V_{GloVe}|}{|V_{Dataset}|}, \tag{1}$$

where $V_{Dataset}$ is the vocabulary of the dataset and V_{GloVe} is the vocabulary in the GloVe embeddings.

3.2 Encoder-Decoder neural networks for taxonomy classifier

Encoder-Decoder Neural Network is a type of neural network that is actively studied in recent years [1, 3, 7], which shows very good performance in various tasks such as machine translation and automatic summarization. We will describe the Encoder-Decoder Neural Network used in this research.

Figure 2 shows our Encoder-Decoder neural network with attention mechanism [1]. Our model has two main functions called encoder and decoder. An encoder function f_{enc} takes an input sequence of words $\mathbf{x} = (x_1, x_2, \dots, x_n)$ and a decoder function f_{dec} predicts the probability of a category path sequence $\mathbf{y} = (y_1, y_2, \dots, y_m)$. f_{enc} outputs a sequnce of hidden states $\mathbf{h} = (h_1, h_2, \dots, h_n)$. To predict y_t, f_{dec} uses information from $\mathbf{h}$ and c_t. A context vector c_t captures input sequence information to help predict an each label y_t. A context vector c_t is defined as following:

$$c_t = \sum_i a_{ti} h_i, \tag{2}$$

and attention is defined as following:

$$a_{ti} = \frac{\hat{a}_{ti}}{\sum_j \hat{a}_{tj}}, \tag{3}$$

$$\hat{a}_{ti} = att(h_i, \bar{h}_t), \tag{4}$$

where $att(h_t, \bar{h}_i)$ is an attention function. The attention function of our works is based on Luong et al. [4] defined as following:

$$att(h_i, \bar{h}_t) = {h_i}^T W_a \bar{h}_t, \tag{5}$$

where h is the encoder state, $\bar{h}$ is the decoder state and W_a is the weight matrix that controls the contribution of each h_i and $\bar{h}_t$.

[1] https://spacy.io

Table 2: Vocabulary matching rate

Preprocessing	The size of source vocabulary	Matching rate
None	670,092	10.69%
lowercase	626,567	57.82%

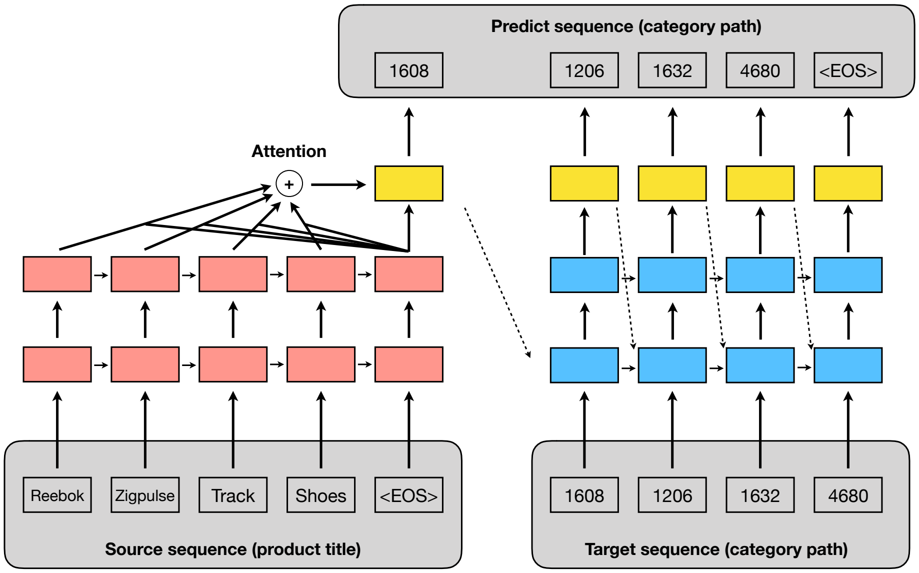
Predict sequence (category path)
1608
1206
1632
4680
<EOS>
Attention
+
Reebok
Zigpulse
Track
Shoes
<EOS>
Source sequence (product title)
1608
1206
1632
4680
Target sequence (category path)

Figure 2: Encoder-Decoder Neural Network

Equation 6 is the log-probability for predicted sequence.

$$\log p(\mathbf{y} \mid \mathbf{x}) = \sum_{t=1}^{n} \log p(y_t \mid \mathbf{y}_{<t}, \mathbf{x}) \tag{6}$$

After the encoder takes input, the decoder predicts outputs using encoder state. As a feature of the Encoder-Decoder neural network, the input sequence length and the output sequence length do not have to match. It can predict various length category path with various length of a description of a product.

4 EXPERIMENTS

This section presents evaluations of our taxonomy classifier and the baseline classifier in the validation set. At the time of training, we use up to 50,000 words as the features both in baseline and the proposed model. In the experiment, we examine parameters of our taxonomy classifier (in Table 3) and show best parameters in each pair of encoder and decoder in Table 4.

4.1 Baseline

We use Random Forest [2] as the baseline. Random Forest is commonly used in various kind of tasks including classification. If we try to solve the multi-label problem where there are 3,695 labels, the computational cost is very expensive. To avoid this difficulty, we use the category path as the label to predict. Therefore our baseline tries to solve the multi-class (3,695 classes) classification problem. We use the TF-IDF vectors for features of the product description representations. To implement the baseline, we use scikit-learn [5]. We use the scikit-learn's default parameters to train the Random Forest.

4.2 Results

We evaluate the performance of our proposed models and the baseline on the validation set with the official script (`eval.py`). We show the best parameters for each model in Table 4, and the results in Table 5. Bidirectional LSTM with GloVe achieves the best F1 score. Our model achieved the best performance when it uses Bidirectional LSTM as an encoder/decoder, lowercase dataset and use GloVe embeddings to initialize the weights of the embedding layer for the input sequence. Interestingly, it shows bad scores when we use GRU for encoder and decoder. We will further investigate the reason for this.

5 CONCLUSION

In this paper, we propose an encoder-decoder neural network for taxonomy classification where there are various sizes of category paths. It is computationally expensive to solve this problem as a multi-label classification because there are over 3,695 categories in the dataset, To avoid this difficulty, we regarded taxonomy classification as the translation from the description of products to the

Table 3: Global training configurations

Parameters	Value
Epoch	150
Optimizer	SGD
Learning rate	1
Learning rate decay	0.99999
Mini batch size	256
Word embedding dim for source (product)	[300, 500]
Pre-trained word embedding	[None, GloVe]
Word embedding dim for target (category)	500
Encoder / Decoder units	[LSTM, BiLSTM, GRU, BiGRU]
Number of Encoder / Decoder	2
Encoder / Decoder dim	[300, 500]
Global attention	Luong et al. [4]
Preprocessing	[None, lowercase]

Table 4: Best parameters for each model

Rnn type	RNN dim	Embedding type	Embedding Dim
BiGRU	300	GloVe	300
BiLSTM	500	GloVe	300
GRU	300	GloVe	300
LSTM	500	GloVe	300

Table 5: Performance comparison. We only show the models which achieved the highest evaluation score in each pairs of encoder and decoder.

Model	RNN dim	Embedding type	Embedding dim	Precision	Recall	F1 score
BiGRU	300	GloVe	300	0.5367	0.5630	0.5292
GRU	300	GloVe	300	0.6031	0.6286	0.5985
Baseline	–	–	–	0.7512	0.7576	0.7476
LSTM	500	GloVe	300	0.8003	0.8001	0.7978
BiLSTM	500	GloVe	300	**0.8024**	**0.8030**	**0.7999**

category path. Experiments show that our approach outperforms Random Forest classifier.

ACKNOWLEDGEMENTS

This work was supported by JSPS KAKENHI Grant Number 16H02904. Also, we would like to show our gratitude to Kento Nozawa and Taro Tezuka for comments that greatly improved the manuscript.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proc. International Conference on Learning Representations*. http://arxiv.org/abs/1409.0473

[2] Leo Breiman. 2001. Random Forests. *Mach. Learn.* 45, 1 (Oct. 2001), 5–32. https://doi.org/10.1023/A:1010933404324

[3] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *Proc. Empirical Methods in Natural Language Processing*. http://emnlp2014.org/papers/pdf/EMNLP2014179.pdfhttp://arxiv.org/abs/1406.1078

[4] Minh-thang Luong and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proc. Empirical Methods in Natural Language Processing*. 1412–1421.

[5] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[6] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proc. Empirical Methods in Natural Language Processing*. 1532–1543. http://www.aclweb.org/anthology/D14-1162

[7] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proc. Advances in Neural Information Processing Systems*. 3104–3112. http://arxiv.org/abs/1409.3215