Label-Guided Learning for Item Categorization in E-commerce

Lei Chen
Rakuten Institute of Technology
Boston, MA
lei.a.chen@rakuten.com

Hirokazu Miyake
Rakuten Institute of Technology
Boston, MA
hirokazu.miyake@rakuten.com

Abstract

Item categorization is an important application of text classification in e-commerce due to its impact on the online shopping experience of users. One class of text classification techniques that has gained attention recently is using the semantic information of the labels to guide the classification task. We have conducted a systematic investigation of the potential benefits of these methods on a real data set from Rakuten, a major e-commerce company in Japan. We found that using pre-trained word embeddings specialized to specific categories of items performed better than one obtained from all available categories despite the reduction in data set size. Furthermore, using a hyperbolic space to embed product labels that are organized in a hierarchical structure led to better performance compared to using a conventional Euclidean space embedding. These findings demonstrate how label-guided learning can improve item categorization systems in the e-commerce domain.

1 Introduction

Natural language processing (NLP) techniques have been applied extensively to solve modern e-commerce challenges (Malmasi et al., 2020; Zhao et al., 2020). One major NLP challenge in e-commerce is *item categorization* (IC) which refers to classifying a product based on textual information, typically the product title, into one of numerous categories in the product category taxonomy tree of online stores. Although significant progress has been made in the area of text classification, many standard open-source data sets have limited numbers of classes which are not representative of data in industry where there can be hundreds or even thousands of classes (Li and Roth, 2002; Pang and Lee, 2004; Socher et al., 2013)To cope with the large number of products and the complexity of the category taxonomy, an automated IC system is needed and its prediction quality needs to be high

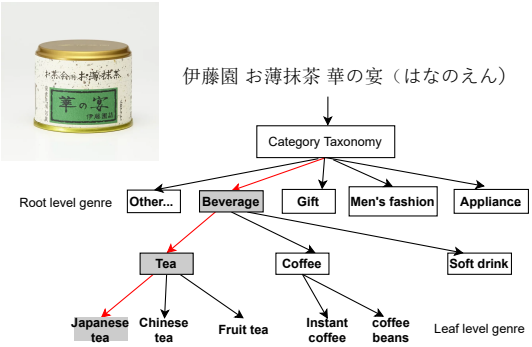


Figure 1: Subset of the product taxonomy tree for item categorization.

enough to provide positive shopping experiences for customers and consequently drive sales. Figure 1 shows an example diagram of the product category taxonomy tree for the IC task. In this example, a tin of Japanese tea [1] needs to be classified into the leaf level category label "Japanese tea."

As reviewed in Section 2, significant progress has been made on IC as a deep learning text classification task. However, much of the progress in text classification does not make use of the semantic information contained in the labels. Recently there have been increasing interest in taking advantage of the semantic information in the labels to improve text classification performance (Wang et al., 2018; Liu et al., 2020; Du et al., 2019; Xiao et al., 2019; Chai et al., 2020). For the IC task, labels in a product taxonomy tree are actively maintained by human experts and these labels bring rich semantic information. For example, descriptive genre information like "clothes" and "electronics" are used rather than just using a numeric index for the class labels. It is reasonable to surmise that leveraging the semantics of these category labels will improve the IC models.

Although label-guided learning has been shown

[1] Image from https://item.rakuten.co.jp/kusurinokiyoshi/10016272/

Proceedings of NAACL HLT 2021: IndustryTrack Papers, pages 296–303
June 6–11, 2021. ©2021 Association for Computational Linguistics

to improve classification performance on several standard text classification data sets, its application to IC on real industry data has been missing thus far. Compared to standard data sets, e-commerce data typically contain more complicated label taxonomy tree structures, and product titles tend to be short and do not use standard grammar. Therefore, whether label-guided learning can help IC in industry or not is an open question worth investigating.

In this paper, we describe our investigation of applying label-guided learning to the IC task. Using real data from Rakuten[2], we tested two models: Label Embedding Attentive Model (LEAM) (Wang et al., 2018) and Label-Specific Attention Network (LSAN) (Xiao et al., 2019). In addition, to cope with the challenge that labels in an IC task tend to be similar to each other within one product genre, we utilized label embedding methods that can better distinguish labels which led to performance gains. This included testing the use of hyperbolic embeddings which can take into account the hierarchical nature of the taxonomy tree (Nickel and Kiela, 2017).

The paper is organized as follows: Section 2 reviews related research on IC using deep learning-based NLP and the emerging techniques of label-guided learning. Section 3 introduces the two label-guided learning models we examined, namely LEAM and LSAN, as well as hyperbolic embedding. Section 4 describes experimental results on a large-scale data set from a major e-commerce company in Japan. Section 5 summarizes our findings and discusses future research directions.

2 Related works

Deep learning-based methods have been widely used for the IC task. This includes the use of deep neural network models for item categorization in a hierarchical classifier structure which showed improved performance over conventional machine learning models (Cevahir and Murakami, 2016), as well as the use of an attention mechanism to identify words that are semantically highly correlated with the predicted categories and therefore can provide improved feature representations for a higher classification performance (Xia et al., 2017).

Recently, using semantic information carried by label names has received increasing attention in text classification research, and LEAM (Wang et al., 2018) is one of the earliest efforts in this direction that we are aware of. It uses a joint embedding of both words and class labels to obtain label-specific attention weights to modify the input features. On a set of benchmark text classification data sets, LEAM showed superior performance over models that did not use label semantics. An extension of LEAM called LguidedLearn (Liu et al., 2020) made further modifications by (a) encoding word inputs first and then using the encoded outputs to compute label attention weights, and (b) using a multi-head attention mechanism (Vaswani et al., 2017) to make the attention-weighting mechanism have more representational power. In a related model, LSAN (Xiao et al., 2019) added a label-specific attention branch in addition to a self-attention branch and showed superior performance over models that did not use label semantics on a set of multi-label text classification tasks.

Alternatively, label names by themselves may not provide sufficient semantic information for accurate text classification. To address this potential shortcoming, longer text can be generated based on class labels to augment the original text input. Text generation methods such as using templates and reinforcement learning were compared, and their effectiveness were evaluated using the BERT model (Devlin et al., 2019) with both text sentence and label description as the input (Chai et al., 2020).

Finally, word embeddings such as word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014) are generated in Euclidean space. However, embeddings in non-Euclidean space called hyperbolic embeddings have been developed (Nickel and Kiela, 2017; Chen et al., 2020a,b) and have been shown to better represent the hierarchical relationship among labels.

3 Model

For a product title $\mathcal{X}$ consisting of L words $\mathcal{X} = [w_1, \ldots, w_L]$, our goal is to predict one out of a set of K labels, $y \in \mathcal{C} = \{c_1, \ldots, c_K\}$. In a neural network-based model, the mapping $\mathcal{X} \rightarrow y$ generally consists of the following steps: (a) *encoding step* (f_0), converting $\mathcal{X}$ into a numeric tensor representing the input, (b) *representation step* (f_1), processing the input tensor to be a fixed-dimension feature vector $\mathbf{z}$, and (c) *classification step* (f_2), mapping $\mathbf{z}$ to y using a feed-forward layer.

Among label-guided learning models, we chose both LEAM (Wang et al., 2018) and LSAN (Xiao

[2] https://www.rakuten.co.jp

Step	LEAM	LSAN
f_0	Word embedding	Word embedding + Bi-LSTM encoding
f_1	Only label-specific attention	Both self- and label-specific attentions + adaptive interpolation
f_2	Softmax with CE loss	Softmax with CE loss

Table 1: Comparison of LEAM (Wang et al., 2018) and LSAN (Xiao et al., 2019) with respect to the three modeling steps.

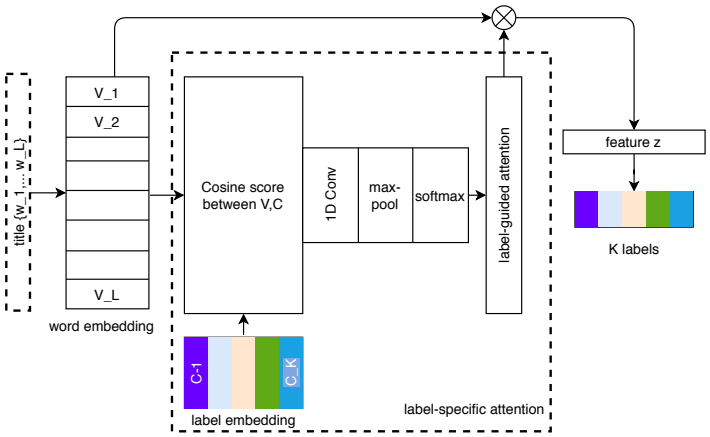


Figure 2: Architecture of LEAM (Wang et al., 2018).

et al., 2019) for our experiments. Table 1 shows a comparison between these models.

3.1 LEAM

The LEAM architecture is shown in Figure 2 (Wang et al., 2018). First a product title of length L is encoded as $V = [v_1, \ldots, v_L]$ where $v_l \in \mathbb{R}^D$ is determined through word embedding and $V \in \mathbb{R}^{D \times L}$. The class labels are also encoded via label embedding as $C = [c_1, ..., c_K]$ where K is the total number of labels, $c_k \in \mathbb{R}^D$ and $C \in \mathbb{R}^{D \times K}$. The label embeddings are title-independent and is the same across all titles for a given set of labels. We can then compute the compatibility of each word-label pair based on their cosine similarity to obtain a compatibility tensor $G \in \mathbb{R}^{L \times K}$.

The compatibility tensor is transformed into an attention vector through the following steps, (a) apply a 1D convolution to refine the compatibility scores by considering its context, (b) apply max pooling to keep the maximum score, and (c) apply a softmax operation to obtain the label-guided attention weights β. These attention weights containing the label semantic information are used in the f_1 step to compute a new representation,

$$z = \sum_l \beta_l v_l. \quad (1)$$

After obtaining z, we use a fully-connected layer with softmax to predict $y \in \mathcal{C}$. The entire process $f_2(f_1(f_0(\mathcal{X})))$ is optimized by minimizing the cross-entropy loss between y and $f_2(z)$.

3.2 LSAN

The LSAN architecture is shown in Figure 3 (Xiao et al., 2019). As shown in Table 1, LSAN has a few modifications compared to LEAM. First, a bi-directional long short-term memory (Bi-LSTM) encoder is used to better capture context semantic cues in the representation. The resulting concatenated tensor is $H = [\overrightarrow{H}, \overleftarrow{H}]$ where $\overrightarrow{H}$ and $\overleftarrow{H}$ represent LSTM encoding outputs from forward and backward directions and $H \in \mathbb{R}^{L \times 2P}$ where P is the dimension of the LSTM hidden state. For model consistency we typically set $P = D$.

Additional features of LSAN which extend LEAM include (a) using self-attention on the encoding H, (b) creating a label-attention component from H and C, and (c) adaptively merging the self- and label-attention components.

More specifically, the self-attention score $A^{(s)}$ is determined as

$$A^{(s)} = \text{softmax}(W_2 \tanh(W_1 H^T)), \quad (2)$$

where $W_1 \in \mathbb{R}^{d_a \times 2P}$ and $W_2 \in \mathbb{R}^{K \times d_a}$ are self-attention tensors to be trained, d_a is a hyperparameter, $A^{(s)} \in \mathbb{R}^{K \times L}$ and each row $A^{(s)}_{j.}$ is an L-dimensional vector representing the contributions

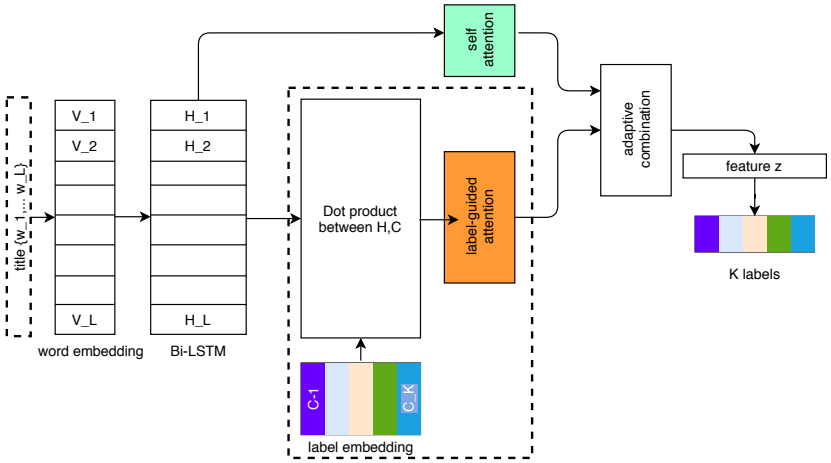


Figure 3: Architecture of LSAN (Xiao et al., 2019).

of all L words to label j. Therefore,

$$M^{(s)} = A^{(s)}H \tag{3}$$

is a representation of the input text weighted by self-attention where $M^{(s)} \in \mathbb{R}^{K\times 2P}$.

From the title encoding H and the label embedding C, compatibility scores between class labels and title words can be computed as the product

$$\overleftarrow{A^{(l)}} = C^T\overleftarrow{H}^T \tag{4}$$

$$\overrightarrow{A^{(l)}} = C^T\overrightarrow{H}^T, \tag{5}$$

where $A^{(l)} \in \mathbb{R}^{K\times L}$ and each row $A^{(l)}_{j\cdot}$ is a L-dimensional vector representing the contributions of all L words to label j. The product title can be represented using label attention as

$$M^{(l)} = [\overleftarrow{A^{(l)}}\overleftarrow{H}, \overrightarrow{A^{(l)}}\overrightarrow{H}] \tag{6}$$

where $M^{(l)} \in \mathbb{R}^{K\times 2P}$.

The last procedure in the f_1 step of LSAN is to adaptively combine the self- and label-attention representations $M^{(s)}$ and $M^{(l)}$ as

$$M_{j\cdot} = \alpha_j M^{(s)}_{j\cdot} + \beta_j M^{(l)}_{j\cdot}, \tag{7}$$

where the two interpolation weight factors ($\alpha, \beta \in \mathbb{R}^K$) are computed as

$$\alpha = \sigma(M^{(s)}W_3) \tag{8}$$

$$\beta = \sigma(M^{(l)}W_4), \tag{9}$$

with the constraint $\alpha_j + \beta_j = 1$, $W_3, W_4 \in \mathbb{R}^{2P}$ are trainable parameters, $\sigma(x) \equiv 1/(1+e^{-x})$ is the element-wise sigmoid function, and $M \in \mathbb{R}^{K\times 2P}$.

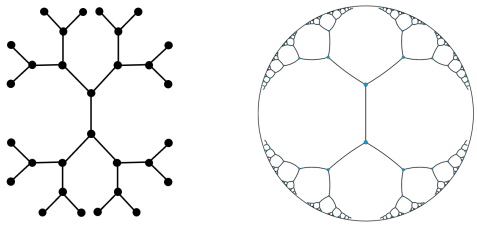
Although the original LSAN model proposed multiple additional layers in its f_2 step, in our implementation we performed average pooling along the label dimension and then to a fully-connected layer with softmax output, similar to LEAM. Finally, the cross entropy loss is minimized.

3.3 Hyperbolic Embedding

In e-commerce item categorization we tend to use a more complicated label structure with a large number of labels organized as a taxonomy tree compared to standard text classification data sets. One immediate issue is that hundreds of labels can exist at the leaf level, some with very similar labels like "Japanese tea" and "Chinese tea," and the difference in label embedding vectors in Euclidean space can be too small to be distinguished by machine learning models. Such issues become more severe with increasing taxonomy tree depth as well. Hyperbolic embedding is one technique that has been developed which can address these issues (Nickel and Kiela, 2017; Chen et al., 2020a,b).

Hyperbolic space is different from Euclidean space by having a negative curvature. Consequently, given a circle, its circumference and disc area grow exponentially with radius. In contrast, in Euclidean space the circumference and area grow only linearly and quadratically, respectively. For representing hierarchical structures like trees, this property can ensure that all leaf nodes which are closer to the edge of the circle maintain large enough distances from each other.

As a specific application, Poincaré embedding uses the Poincaré ball model which consists of points within the unit ball $\mathbb{B}^d$ where the distance



(a) (b)

Figure 4: (a) Tree with a branching factor of 2 in Euclidean space. (b) Embedding a hierarchical tree with a branching factor of 2 in a Poincaré disk. Figure from Figure 1(b) in (Nickel and Kiela, 2017).

between two points, $\mathbf{u}, \mathbf{v} \in \mathbb{B}^{\mathbf{d}}$ is defined as

$$d(\mathbf{u}, \mathbf{v}) = \cosh^{-1}\left(1+2\frac{\|\mathbf{u}-\mathbf{v}\|^2}{(1-\|\mathbf{u}\|^2)(1-\|\mathbf{v}\|^2)}\right). \quad (10)$$

The Poincaré embedding is obtained by minimizing a loss function depending only on $d(\mathbf{u}, \mathbf{v})$ for all pairs of labels $(\mathbf{u}, \mathbf{v})$ using Riemannian optimization methods.

Figure 4 illustrates the differences between using an Euclidean space and a Poincaré ball model when representing nodes organized in a tree. Using a hyperbolic embedding has the potential to maintain large enough distances when our models aim to distinguish subtle differences among these labels.

4 Experiments and Results

4.1 Experimental Setup

Data set: Our data set consisted of more than one million products in aggregate from Rakuten, a large e-commerce platform in Japan, focusing on four major product categories which we call root-level genres. Our task, a multi-class classification problem, was to predict the leaf-level product categories from their Japanese titles. Further details of our data set are shown in Table 2.

Evaluation metric: We used the macro-averaged F-score F to evaluate model performance. This is defined in terms of the per-class F-score F_k as

$$F = \frac{1}{K}\sum_{k=1}^{K} F_k, \quad (11)$$

$$F_k = \frac{2P_kR_k}{P_k + R_k}, \quad (12)$$

where K is the total number of classes, and P_k and R_k are the precision and recall for class k.

Pre-trained embedding methods: We tested the following three methods:

- All genre: Word embedding pre-trained on all of the data across different root-level genres; for the label embedding, the average of the word embedding from all word tokens in a label is used to initialize the label embedding C and this is further updated in the model training process.
- Genre specific: Word embedding pre-trained from data specific to each root-level genre; label embeddings were obtained similarly to the all-genre method.
- Poincaré: Label embedding pre-trained on the Poincaré ball taking into account the full hierarchical taxonomy tree.

Models: We compared a number of variants of LEAM and LSAN as described below.

- LEAM: Described in Section 3 (Wang et al., 2018), using all-genre pre-trained word embeddings.
- $LEAM_{\text{base}}$: LEAM without the label embedding attention component (effectively fixing $\beta_l = 1/L$ in Eq. 1), using all-genre pre-trained word embeddings.
- LSAN: Described in Section 3 (Xiao et al., 2019), using all-genre pre-trained word embeddings.
- $LSAN_{\text{base}}$: LSAN without the label-specific attention component (effectively fixing $\beta = 0$ in Eq. 7) which is similar to AttentionXML (You et al., 2019), and using all-genre pre-trained word embeddings.
- $LSAN_{\text{genre}}$: LSAN using genre-specific pre-trained word embeddings.
- $LSAN_{\text{Poincaré}}$: LSAN using genre-specific pre-trained word embeddings for the titles and pre-trained Poincaré embeddings for the labels.

Experimental parameters: Our models were implemented in TensorFlow 2.3 using a GPU for training and evaluation. Since Japanese text does not have spaces to indicate individual words, tokenization was performed with MeCab, an open source

Root genre	Class size	Train size	Dev size	Test size	Mean words/title
Catalog Gifts & Tickets	29	11,369	1,281	559	31
Beverages	32	205,107	22,805	10,315	21
Appliances	286	399,584	44,529	18,478	20
Men's Fashion	71	593,126	65,939	43,243	23

Table 2: Summary of our data set obtained from a large e-commerce platform in Japan.

Root genre	$LEAM_{\text{base}}$	LEAM	$LSAN_{\text{base}}$	LSAN
Catalog Gifts & Tickets	0.341	0.289↓	0.241	0.338
Beverages	0.719	0.755	0.759	0.773
Appliances	0.682	0.654↓	0.667	0.686
Men's Fashion	0.696	0.657↓	0.685	0.686

Table 3: Macro F-score of LEAM and LSAN without and with label attention.

Japanese part-of-speech and morphological analyzer using conditional random fields (CRF).[3] Once the text was tokenized, we fixed our input length to $L = 60$ words by truncating the title if it was longer than L and zero-padding the title if it was shorter than L. If a word appeared less than three times, it was discarded and replaced with an out-of-vocabulary token.

Pre-trained word embeddings of dimension $D = 100$ using just product titles were obtained with fastText, which uses a skipgram model with bag-of-character n-grams (Bojanowski et al., 2016). No external pre-trained embeddings were used. After initialization of word and label embeddings with pre-trained values, they were jointly trained with the remaining parameters of the model.

For Poincaré embedding of labels, we used an embedding dimension of 300. Pre-trained Poincaré embeddings of labels were obtained by representing the genre taxonomy tree as (child, parent) pairs and minimizing a loss function which depends only on inter-genre distances as defined in Eq. 10 (Nickel and Kiela, 2017). These pre-trained Poincaré label embeddings were used to initialize the label embeddings in LSAN but during training were allowed to vary according to the standard loss optimization process in Euclidean space.

For LEAM, we used a 1D convolution window size of 5. For LSAN, we set $d_a = 50$, and when we experimented with the Poincaré embedding we set the LSTM hidden state dimension $P = 300$ to match the Poincaré embedding dimension.

The models were trained by minimizing the cross-entropy loss function using the Adam optimizer with an initial learning rate of 0.001 (Kingma and Ba, 2015). We used early stopping with a patience of 10 to obtain the final models.

4.2 Results and Discussions

Impact of label attention: We examined the impact of label attention by comparing performance without and with label attention for LEAM and LSAN for each of the four root-level genres using all-genre pre-trained word embeddings. The result is shown in Table 3. For LEAM, we do not observe consistent improvements by including the label attention component, contrary to what was previously reported on standard text classification data sets (Wang et al., 2018). On the other hand for LSAN we do observe consistent improvements over all root-level genres by including the label attention component of the model. Since we did not observe a consistent improvement for LEAM in using label attention, for the remainder of this section we focus on variations of LSAN.

Impact of different pre-trained embeddings: We next evaluated the impact of using different pre-trained embeddings for the title embeddings as well as the label embeddings for each of the four root-level genres. This is shown in Table 4. We observed that different pre-trained embeddings can consistently have a significant effect on model performance. In particular, using genre-specific embeddings outperformed all-genre embeddings for all genres. This is particularly notable for the smallest genre where we used more than 10 times the data to obtain the all-genre embeddings.

We believe this is because words that occur in the same root-level genre will tend to be embedded closer to each other in the full embedding space,

[3] https://taku910.github.io/mecab/

Root genre	LSAN	$LSAN_{\text{genre}}$	$LSAN_{\text{Poincaré}}$
Catalog Gifts & Tickets	0.338	0.403	**0.438**
Beverages	0.773	0.784	**0.789**
Appliances	0.686	0.697	**0.701**
Men's Fashion	0.686	0.701	**0.722**

Table 4: Macro F-score of LSAN with various pre-trained title and label embeddings.

which then makes it more difficult for the label attention to distinguish between different but similar labels such as "Japanese tea" and "Chinese tea." By using pre-trained embeddings obtained from specific genres, the embeddings become spaced farther apart and therefore the label attention is able to better distinguish labels with similar names.

Poincaré embeddings take this further by requiring the embedding space distance between all leaf-genre labels to be far apart from each other, and our results show that this leads to the best model performance. This supports our hypothesis that the distance between labels in the label embedding space is an important factor in ensuring that label attention improves model performance.

Compared to models using only the product titles, we see that models using label-guided learning can significantly improve the F-score. In particular, LSAN using a Poincaré label embedding shows the following F-score increases compared to LSAN base: 19.7% for "Catalog Gifts & Tickets," 3.0% for "Beverages," 3.4% for "Appliances," and 3.7% for "Men's Fashion." Note that the largest increase was achieved on the genre with the fewest training instances.

5 Conclusions

Since 2018, there have been increasing interest in the field of NLP to use the semantic information of class labels to further improve text classification performance. On the item categorization task in e-commerce, a taxonomy organized in a hierarchical structure already contains rich meaning and provides an ideal opportunity to evaluate the impact of label-guided learning. In this paper, we used real industry data from Rakuten, a leading Japanese e-commerce platform, to evaluate the benefits of label-guided learning.

Our experiments showed that LSAN is superior to LEAM because of its usage of context encoding and adaptive combination of both self- and label-attention. We also found that using genre-specific pre-trained embeddings led to better model performance than pre-trained embeddings obtained from all product genres. This is likely because pre-training on specific genres allows the embedding to focus on differences between similar genres and the label embeddings are able to take advantage of this. Finally, we showed that using hyperbolic embedding, more specifically Poincaré embedding, can improve model performance further by ensuring that all class labels are sufficiently separated to allow label-guided learning to work well.

One possible limitation of our current work is that although the label embedding is initialized using a hyperbolic embedding, the rest of the training process proceeds in Euclidean space. Future work could explore the possibility of training the entire model in hyperbolic space. Another direction is to incorporate the label-attention mechanism into the BERT model (Devlin et al., 2019), which has proven to be a powerful approach to text encoding. In addition, more advanced approaches to obtaining better representations of labels on top of our existing approach of using word tokens in labels could be explored.

Acknowledgements

The authors would like to thank Yandi Xia for introducing hyperbolic embeddings to us and pre-training the Poincaré embeddings.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Ali Cevahir and Koji Murakami. 2016. Large-scale multi-class and hierarchical product categorization for an E-commerce giant. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 525–535, Osaka, Japan. The COLING 2016 Organizing Committee.

Duo Chai, Wei Wu, Qinghong Han, Fei Wu, and Jiwei Li. 2020. Description based text classification with reinforcement learning. In *Proceedings of the 37th*

International Conference on Machine Learning, volume 119 of *Proceedings of Machine Learning Research*, pages 1371–1382. PMLR.

Boli Chen, Xin Huang, Lin Xiao, Zixin Cai, and Liping Jing. 2020a. Hyperbolic interaction model for hierarchical multi-label classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7496–7503.

Boli Chen, Xin Huang, Lin Xiao, and Liping Jing. 2020b. Hyperbolic capsule networks for multi-label classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3115–3124, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*. ArXiv: 1810.04805.

Cunxiao Du, Zhaozheng Chen, Fuli Feng, Lei Zhu, Tian Gan, and Liqiang Nie. 2019. Explicit interaction model towards text classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6359–6366.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Xien Liu, Song Wang, Xiao Zhang, Xinxin You, Ji Wu, and Dejing Dou. 2020. Label-guided Learning for Text Classification. *arXiv preprint arXiv:2002.10772*.

Shervin Malmasi, Surya Kallumadi, Nicola Ueffing, Oleg Rokhlenko, Eugene Agichtein, and Ido Guy, editors. 2020. *Proceedings of The 3rd Workshop on e-Commerce and NLP*. Association for Computational Linguistics, Seattle, WA, USA.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278, Barcelona, Spain.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. 2018. Joint embedding of words and labels for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2321–2331, Melbourne, Australia. Association for Computational Linguistics.

Yandi Xia, Aaron Levine, Pradipto Das, Giuseppe Di Fabbrizio, Keiji Shinzato, and Ankur Datta. 2017. Large-scale categorization of Japanese product titles using neural attention models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 663–668, Valencia, Spain. Association for Computational Linguistics.

Lin Xiao, Xin Huang, Boli Chen, and Liping Jing. 2019. Label-specific document representation for multi-label text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 466–475, Hong Kong, China. Association for Computational Linguistics.

Ronghui You, Zihan Zhang, Ziye Wang, Suyang Dai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2019. Attentionxml: Label tree-based attention-aware deep model for high-performance extreme multi-label text classification. In *Advances in Neural Information Processing Systems*, volume 32, pages 5820–5830. Curran Associates, Inc.

Huasha Zhao, Parikshit Sondhi, Nguyen Bach, Sanjika Hewavitharana, Yifan He, Luo Si, and Heng Ji, editors. 2020. *Proceedings of Workshop on Natural Language Processing in E-Commerce*. Association for Computational Linguistics, Barcelona, Spain.