Label Embedding using Hierarchical Structure of Labels for Twitter Classification

Taro Miyazaki **Kiminobu Makino** **Yuka Takei** **Hiroki Okamoto** **Jun Goto**
NHK Science and Technology Research Laboratories
{miyazaki.t-jw, makino.k-gg, takei.y-ek,
okamoto.h-iw, goto.j-fw}@nhk.or.jp

Abstract

Twitter is used for various applications such as disaster monitoring and news material gathering. In these applications, each Tweet is classified into pre-defined classes. These classes have a semantic relationship with each other and can be classified into a hierarchical structure, which is regarded as important information. Label texts of pre-defined classes themselves also include important clues for classification. Therefore, we propose a method that can consider the hierarchical structure of labels and label texts themselves. We conducted evaluation over the Text REtrieval Conference (TREC) 2018 Incident Streams (IS) track dataset, and we found that our method outperformed the methods of the conference participants.

1 Introduction

Twitter is used for various applications such as disaster monitoring (Ashktorab et al., 2014; Mizuno et al., 2016) and news material gathering (Vosecky et al., 2013; Hayashi et al., 2015). In these applications, each Tweet is classified into pre-defined classes. These classes have a semantic relationship with each other and can be classified into a hierarchical structure. Consider a news material gathering system that has classes such as "Tornado," "Flood," and "Riot." These can be classified into two upper classes, "*Natural disaster*" ("Tornado" and "Flood" as lower classes) and "*Incident*" ("Riot" as a lower class). These relationships can be an important clue for classification. Needless to say, label texts of pre-defined classes themselves include important clues. People can understand what the criterion of the classification is by reading labels. Therefore, we propose a method that can consider the hierarchical structure of labels and the labels themselves.

Our method is based on the Label Embedding (LE) method (Zhang et al., 2018). The typical LE method embeds input text and label text, and then the embedded vectors are fed into "Matcher," which outputs the score between the input text and label. We use a two-step attention mechanism for LE to consider the hierarchical structure of labels. We confirm the effectiveness of our method through evaluation over the Text REtrieval Conference (TREC) 2018 Incident Streams (IS) track dataset.

Our contributions are as follows: (1) we propose label embedding using the hierarchical structure of labels, and (2) our method outperformed others on the TREC 2018 IS track dataset for several metrics, including the Any-type Micro F1 score, which is the target metric of the track.

2 TREC 2018 IS track

The TREC 2018 IS track is a shared-task that aims to mature social media-based emergency response technology (McCreadie et al., 2019). The task of the track is classifying Tweets by information type, which consists of 24 classes. The list of information types and the description for each information type are given as ontology as shown in Table 1. The dataset contains approximately 1,300 Tweets for training and more than 20,000 for testing. The numbers of samples for each class are also given in the table[1]. As shown in the table, a training dataset does not include much data and is unbalanced for classes.

The track uses two evaluation methods — Multi-type and Any-type — as follows[2].

Multi-type: Calculating categorization performance per information type in a 1-vs-All manner.

[1] Note that, each sample has just one information type for training data, but more than one for testing data.

[2] See http://trecis.org/2018/Evaluation.html for details.

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 6317–6322,
Hong Kong, China, November 3–7, 2019. ©2019 Association for Computational Linguistics

Table 1: Ontology information type.

Intent type	Information type	Description	# (train)	# (test)
Request	Request-GoodsServices	The user is asking for a particular service or physical good	0	126
	Request-SearchAndRescue	The user is requesting a rescue (for themselves or others)	0	286
	Request-InformationWanted	The user is requesting information	10	172
Report	Report-Weather	The user is providing a weather report (current or forecast)	41	1,325
	Report-FirstPartyObservation	The user is giving an eye-witness account	28	3,807
	Report-ThirdPartyObservation	The user is reporting a information that they received from someone else	15	4,160
	Report-EmergingThreats	The user is reporting a potential problem that may cause future loss of life or damage	36	686
	Report-SignificantEventChange	The user is reporting a new occurrence that public safety officers need to respond to	34	415
	Report-MultimediaShare	The user is sharing images or video	127	3,974
	Report-ServiceAvailable	The user is reporting that they or someone else is providing a service	15	1,076
	Report-Factoid	The user is relating some facts, typically numerical	140	2,383
	Report-Official	An official report by a government or public safety representative	52	403
	Report-CleanUp	A report of the clean up after the event	2	62
	Report-Hashtags	Reporting which hashtags correspond to each event	4	3,363
CallToAction	CallToAction-Volunteer	The user is asking people to volunteer to help the response effort	2	116
	CallToAction-Donations	The user is asking people to donate goods/money	15	804
	CallToAction-MovePeople	The user is asking people to leave an area or go to another area	26	27
Other	Other-PastNews	The post is generic news, e.g. reporting that the event occurred	12	1,351
	Other-ContinuingNews	The post providing/linking to continuous coverage of the event	250	4,871
	Other-Advice	The author is providing some advice to the public	39	1,209
	Other-Sentiment	The post is expressing some sentiment about the event	39	6,952
	Other-Discussion	Users are discussing the event	51	2,060
	Other-Irrelevant	The post is irrelevant, contains no information	163	2,605
	Other-Unknown	Does not fit into any other category	26	77
	Other-KnownAlready	The responder already knows this information	112	1101

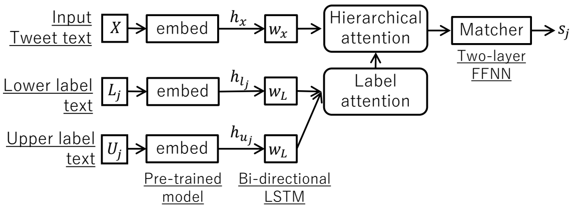

Figure 1: Overview of our proposed method.

Methods are evaluated using four metrics: Precision, Recall, F1 (positive class), and Accuracy (overall).

Any-type: A system receives a full score for a Tweet if the system assigned any of the categories that the human assessor selected for that Tweet. Methods are evaluated using four metrics: Precision, Recall, F1 (micro average: the target metric of the track), and Accuracy (micro average).

3 Our Method

As shown in Table 1, each information type has a rather rich description, so we think that using the description can improve the performance. Also, the information types can be regarded as a hierarchical-structure label, which consists of four upper classes (equal to intent type, such as Request and Report) and 24 lower classes (equal to information type, such as Request-GoodsServices and Report-Weather). To use these useful features, we propose a label embedding using the hierarchical structure of labels.

The overview of our method is given in Figure 1. Our method is similar to the one Zhang et al. (2018) used, but it differs in that our method considers the hierarchical structure of labels, which is our key feature.

X, L_j, and U_j are the input text, label text of the j-th lower class, and label text of the upper class corresponding to the j-th lower class, respectively. We use "description" and "intent type" in Table 1 for the label text of the lower and upper classes, respectively. "Embed" in the figure is a pre-trained model such as BERT (Devlin et al., 2019) and ELMo (Peters et al., 2018) of $V \times m$, where V and m denote the vocabulary size and embedding size, respectively. $\mathbf{w}_X$ and $\mathbf{w}_L$ means Bi-directional Long Short-Term Memory (Bi-LSTM), and both have a d dimension of hidden layer.

First, X, L_j, and U_j are embedded using the pre-trained model, and then the embedded vectors are fed into $\mathbf{w}_X$ for X and $\mathbf{w}_L$ for L_j and U_j, respectively. We gather the hidden states of Bi-LSTM for each input token and stack them, and then we get three matrices:

$$\begin{aligned}
\text{Text representing matrix: } \mathbf{h}_x &\in \mathbb{R}^{|X| \times 2d}, \\
\text{Lower label representing matrix: } \mathbf{h}_{l_j} &\in \mathbb{R}^{|L_j| \times 2d}, \\
\text{Upper label representing matrix: } \mathbf{h}_{u_j} &\in \mathbb{R}^{|U_j| \times 2d},
\end{aligned}$$

where $|X|$, $|L_j|$ and $|U_j|$ represent the number of tokens in X, L_j and U_j, respectively. Note that we have a $2d$ dimension for the row of each matrix because of concatenating forward and backward states.

Next, we calculate "label attention" and "hierarchical attention." The label attention weight $\mathbf{w}_{\text{label}} \in \mathbb{1}^{|L_j|}$ and label attention vector $\mathbf{a}_{\text{label}} \in$

$\mathbb{1}^{2d}$ are calculated as follows:

$$\mathbf{w}_{\text{label}} = \text{softmax}\left(\mathbf{h} \cdot \mathbf{h}_{l_j}^T / \sqrt{2d}\right),$$
$$\mathbf{a}_{\text{label}} = \mathbf{w}_{\text{label}} \cdot \mathbf{h}_{l_j},$$

where $\mathbf{h} \in \mathbb{1}^{2d}$ is the concatenation of hidden states of the final step of both directions of $\mathbf{h}_{u_j}$. We use scaled-dot product attention for label attention (Vaswani et al., 2017).

By using $\mathbf{a}_{\text{label}}$, the hierarchical attention weight $\mathbf{w}_{\text{hier}} \in \mathbb{1}^{|X|}$ and hierarchical attention vector $\mathbf{a}_{\text{hier}} \in \mathbb{1}^{2d}$ are obtained as follows:

$$\mathbf{w}_{\text{hier}} = \mathbf{a}_{\text{label}} \cdot \mathbf{h}_x^T,$$
$$\mathbf{a}_{\text{hier}} = \mathbf{w}_{\text{hier}} \cdot \mathbf{h}_x.$$

Note that we use normalization for label attention but not for hierarchical attention. We confirm that this condition is best by an experiment, which is detailed in the Discussion section.

Then, $\mathbf{a}_{\text{hier}}$ is fed to "Matcher," which consists of a two-layer Feed-Forward Neural Network (FFNN):

$$\mathbf{v}_{\text{mid}} = \mathbf{W}_{\text{mid}} \cdot \text{ReLU}\left(\mathbf{a}_{\text{hier}}\right) + \mathbf{b}_{\text{mid}},$$
$$s_j = \mathbf{W}_{\text{matcher}} \cdot \text{ReLU}\left(\mathbf{v}_{\text{mid}}\right) + \mathbf{b}_{\text{matcher}},$$

where $\mathbf{W}_{\text{mid}} \in \mathbb{R}^{d \times 2d}$ and $\mathbf{W}_{\text{matcher}} \in \mathbb{R}^{1 \times d}$ are weight matrices, and $\mathbf{b}_{\text{mid}} \in \mathbb{1}^d$ and $\mathbf{b}_{\text{matcher}} \in \mathbb{1}^1$ are bias. s_j is a score between input Tweet X and information type j.

We gather s_j for all $j \in C$, where C denotes the set of the information type. Finally, we obtain the estimation result o for input Tweet X as follows:

$$o = \text{argmax}\left(s_1, s_2, \ldots, s_{|C|}\right).$$

4 Experiments

4.1 Experimental Settings

Our experiments were based on TREC 2018 IS track in terms of the dataset and evaluation. The dataset consists of original json data of Tweets. We excluded @-mentions and URLs. We conducted 10-fold cross validation to find the best setting, and all models were used as ensemble models to predict the information type for testing data.

The models were implemented in Chainer (Tokui et al., 2015) and learned with the Adam optimizer (Kingma and Ba, 2014), on the basis of categorical cross-entropy loss with class weights $W_c = \frac{|c_{max}|}{|c|}$, where $|c|$ is the number of samples of information type c appearing in the training data, and $|c_{max}|$ is that of the most-frequent information type. We used BERT-base, Uncased[3] as the pre-trained model, which has a 12-layer, 768-hidden states transformer model with 12-heads attention. The BERT model was used as a frozen model without fine-tuning.

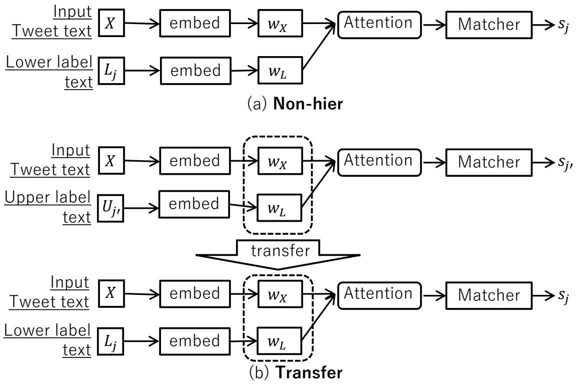

Figure 2: Baseline methods based on label embedding.

The hyperparameters used were: a minibatch size of 32; hidden layer size of 200; L_2 regularization coefficient of 10^{-5}; dropout rate of 0.5; and 100 training iterations, with early stopping on the basis of the Macro F1 score of the development data.

4.2 Baseline Methods

We prepared three baseline methods. All baseline methods used the same hyperparameters as the proposed method.

Non-LE: This method is not based on LE. We use the BERT model as token embedding. The embedded vectors are fed into Bi-LSTM, and then the concatenation of hidden states of the final step of both directions is fed into the two-layer FFNN for classification into the information type.

Non-hier: This method is almost the same as our proposed method but does not use hierarchical attention. The vector fed into Matcher is calculated as attention between embedding vectors of the input text and label text of the lower class (Figure 2-(a)).

Transfer: This method uses the same structure as Non-hier, but we use transfer learning to consider the hierarchical structure of labels. This is inspired by Shimura et al. (2018). First, the model is learned as classifying upper class labels, and then the model is transferred to be learned as classifying lower class labels (Figure 2-(b)). We tried several settings and found that transferring only the Bi-LSTM layer is better in this task.

[3] https://github.com/google-research/bert

Table 2: Experimental results. *italics* denote the best results in TREC 2018, and **bold** denotes the overall best results. For some metrics, the best results are for other TREC 2018 participants' methods not shown in the table, so no results are in *italics* or **bold** for these metrics.

Method	Multi-type, Macro Avg.				Any-type, Micro Avg.			
	Precision	Recall	F1	Accuracy	Precision	Recall	F1 (target metric)	Accuracy
Non-LE	0.2461	0.1302	0.1523	0.9073	0.5032	0.7055	0.5874	0.4387
Non-hier	0.2522	0.1300	0.1497	0.9065	0.5206	0.6301	0.5701	0.4282
Transfer	0.2448	0.1324	0.1533	0.9071	0.5097	0.6810	0.5830	0.4361
Proposed	0.2464	0.1300	0.1521	**0.9079**	0.5164	0.7018	**0.5950**	**0.4460**
TREC2018 participants (Top-3 on Any-type Micro F1 score and Top-1 on Multi-type Macro F1 score)								
cbunS2	***0.2666***	0.1122	0.1262	*0.9059*	0.4549	0.7780	*0.5749*	*0.4213*
KDEIS4_DM	0.1483	0.0708	0.0734	0.9035	0.3914	0.9856	0.5603	0.3908
umdhcilfasttext	0.1827	0.0962	0.1117	0.9044	0.4534	0.7260	0.5582	0.4022
DLR_Augmented	0.2496	0.1657	***0.1571***	0.8996	0.3965	0.6165	0.4826	0.3419

4.3 Results

Table 2 presents the results for our methods and part of the TREC 2018 participants' results. At TREC 2018, cbunS2, KDEIS4_DM, and umdhcilfasttext had the top three results for the Any-type Micro F1 score, which is the target metric of the track. DLR_Augmented had the best Multi-type Macro F1 score.

Our proposed method outperformed others in terms of the Any-type Micro F1 score and both types of accuracy, while the best for the Multi-type F1 score is DLR_Augmented.

4.4 Discussion

Comparison with other methods

Our proposed method performs the best for several metrics including the target metric (Any-type Micro F1 score). Our hierarchical attention can give less weight for verbose phrases such as "The user is asking" in label text. These phrases include only a little information useful for classifying, so giving them less weight is reasonable and effective. This is one of the advantages of our method, which distinguishes it from LE-based baseline methods. Our method outperforms not only LE-based methods, but also the simple but strong baseline (Non-LE) and the TREC 2018 participants' methods. Therefore, we confirmed the effectiveness of our proposed method.

On the other hand, our method did not perform the best for the Multi-type Macro F1 score. DLR_Augmented, which performed the best for this metric, uses augmented data. This is effective in the learning process especially for classes that have small training data. Our method does not use augmented data, so it has lower accuracy for these classes than DLR_Augmented. Comparing our proposed method with the baseline methods, differences in the Multi-type Macro F1 score were small, and the proposed method achieved the best score for Multi-type accuracy. We think that the differences come from labels that have a small number of samples in test set because the F1 scores for these classes are sensitive to the output, which greatly affects the Macro F1 score.

Table 3: Comparing normalization methods.

label attention	hierarchical attention	Macro F1 score
normalized	normalized	0.2676
normalized	**none**	**0.3539**
none	normalized	0.0303
none	none	0.0329

Insight of our proposed method

To determine which combination of normalization is effective for our task, we conducted a small experiment using development data. Results are shown in Table 3. We used the Macro F1 score as the metric[4]. We can see that using normalization for label attention but not for hierarchical attention is best. One reason is that the norm of the output vector of hierarchical attention directly affects the score s_j. Hierarchical attention without normalization can make distinctions for the output vector of each class in the norm, so it works well. On the other hand, using normalization is better in label attention. We observed that not using normalization for label attention makes the model diverge, so normalization needs to be used for label attention.

Table 4 shows the macro F1 scores summarized for each upper class of the proposed method and Non-hier. Δ in the table means the difference between two methods. Interestingly, our method works best for the upper class OTHER, which contains the fewest meanings in the upper class label

[4] Each sample has just one information type for development data and more than one for testing data, so the Macro F1 score between Table 2 and 3 cannot be compared directly.

Table 4: Comparing macro F1 score for each upper class between proposed and Non-hier.

Upper class	Proposed	Non-hier	Δ
Request	0.030	0.037	-0.007
Report	0.124	0.125	-0.001
CallToAction	0.180	0.170	+0.010
Other	0.228	0.216	+0.012

text. The lower class label text for the upper class Other includes less verbose phrases such as "The user requesting," so label attention can give higher weight for important parts of the label text more precisely.

On the other hand, the F1 score of the proposed method for the upper class Report is worse than that of Non-hier. This is caused by the lower class Report-SignificantEventChange (the Δ is -0.05, which is the worst in all lower classes). The label text of the lower class includes a misspelling ("occurence" instead of "occurrence"[5]). This confuses our model when calculating label attention weight, so the F1 score is lower.

We found that the number of tokens in the lower class label text affects the effectiveness of the proposed method. The proposed method has a better F1 score than Non-hier for nine lower classes with a mean number of tokens in the lower class labels of 11.2. On the other hand, the proposed method has a lower F1 score than Non-hier for seven lower classes with a mean number of tokens of 8.0. This shows that the more tokens the lower class label text has, the more effective the proposed method becomes. Of course, this does not mean that our method works for only classes that have rich label text. For example, our method improves the F1 score for some classes that have short label text such as Other-Discussion, which has only five tokens in the label text.

Overall, using hierarchical structure of labels is effective in many cases, but it is sensitive to the quality and quantity of label texts of the lower classes.

5 Related Work

Label embedding has attracted attention, especially for few- and zero-shot learning tasks (Socher et al., 2013; Akata et al., 2013; Kodirov et al., 2017). Now, many methods applied to natural language processing are proposed. Zhang et al. (2018) used multi-task learning, and Xia et al. (2018) used capsule networks for LE and obtained good results. Our method differs in that it considers the hierarchical structure of labels.

There are many studies using a pre-defined hierarchical structure of labels (Wu et al., 2014; Li et al., 2015; Bilal et al., 2018), and some methods are used with LE (Bengio et al., 2010; Ren et al., 2016). These approaches are effective but large-scale hierarchical data from a thesaurus need to be prepared. Our method does not need large-scale data, so it is advantageous when it comes to calculating cost and flexibility. Shimura et al. (2018) used transfer learning to consider the hierarchical structure of labels, which is the basis of our baseline method (Transfer).

6 Conclusion and Future Work

In this paper, we proposed a method of label embedding using the hierarchical structure of labels and confirmed its effectiveness through evaluation over the Text REtrieval Conference (TREC) 2018 Incident Streams (IS) track dataset. Our method outperformed other methods and obtained the best results on the dataset for several metrics including the Any-type Micro F1 score, which was the target metric of TREC 2018 IS track.

We used rather long sentences for lower label text. Typical applications use shorter labels such as "*Tornado*" and "*Riot*." Confirming whether our method works well for other datasets including ones that have only shorter label texts is left for our future work.

Acknowledgements

We would like to thank Prof. Timothy Baldwin and Dr. Ichiro Yamada for valuable discussions, and the TREC 2018 IS track organizers for providing the dataset. We also thank the anonymous reviewers for their helpful comments.

References

Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid. 2013. Label-embedding for attribute-based classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 819–826.

Zahra Ashktorab, Christopher Brown, Manojit Nandi, and Aron Culotta. 2014. Tweedr: Mining twitter to

[5] Our label texts are made from provided ontology, so they contain misspellings arising from the original ontology.

inform disaster response. In *Proceedings of the 11th International ISCRAM Conference*.

Samy Bengio, Jason Weston, and David Grangier. 2010. Label embedding trees for large multi-class tasks. In *Advances in Neural Information Processing Systems 23*, pages 163–171.

Alsallakh Bilal, Amin Jourabloo, Mao Ye, Xiaoming Liu, and Liu Ren. 2018. Do convolutional neural networks learn class hierarchy? *IEEE transactions on visualization and computer graphics*, 24(1):152–162.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4171–4186.

Kohei Hayashi, Takanori Maehara, Masashi Toyoda, and Ken-ichi Kawarabayashi. 2015. Real-time top-r topic detection on twitter with topic hijack filtering. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 417–426.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Elyor Kodirov, Tao Xiang, and Shaogang Gong. 2017. Semantic autoencoder for zero-shot learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3174–3183.

Xirong Li, Shuai Liao, Weiyu Lan, Xiaoyong Du, and Gang Yang. 2015. Zero-shot image tagging by hierarchical semantic embedding. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 879–882.

Richard McCreadie, Cody Buntain, and Ian Soboroff. 2019. TREC incident streams: Finding actionable information on social media. In *Proceedings of 16th International Conference on Information Systems for Crisis Response and Management*.

Junta Mizuno, Masahiro Tanaka, Kiyonori Ohtake, Jong-Hoon Oh, Julien Kloetzer, Chikara Hashimoto, and Kentaro Torisawa. 2016. WISDOM X, DISAANA and D-SUMM: Large-scale NLP systems for analyzing textual big data. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pages 263–267.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 2227–2237.

Xiang Ren, Wenqi He, Meng Qu, Lifu Huang, Heng Ji, and Jiawei Han. 2016. Afet: Automatic fine-grained entity typing by hierarchical partial-label embedding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1369–1378.

Kazuya Shimura, Jiyi Li, and Fumiyo Fukumoto. 2018. HFT-CNN: Learning hierarchical category structure for multi-label short text categorization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 811–816.

Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. 2013. Zero-shot learning through cross-modal transfer. In *Advances in neural information processing systems*, pages 935–943.

Seiya Tokui, Kenta Oono, Shohei Hido, and Justin Clayton. 2015. Chainer: a next-generation open source framework for deep learning. In *Proceedings of workshop on machine learning systems (LearningSys) in the twenty-ninth annual conference on neural information processing systems (NIPS)*, volume 5, pages 1–6.

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 *Advances in neural information processing systems*, pages 5998–6008.

Jan Vosecky, Di Jiang, Kenneth Wai-Ting Leung, and Wilfred Ng. 2013. Dynamic multi-faceted topic discovery in twitter. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 879–884.

Chenxia Wu, Ian Lenz, and Ashutosh Saxena. 2014. Hierarchical semantic labeling for task-relevant RGB-D perception. In *Robotics: Science and systems*.

Congying Xia, Chenwei Zhang, Xiaohui Yan, Yi Chang, and Philip Yu. 2018. Zero-shot user intent detection via capsule neural networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3090–3099.

Honglun Zhang, Liqiang Xiao, Wenqing Chen, Yongkun Wang, and Yaohui Jin. 2018. Multi-task label embedding for text classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4545–4553.