Extracting Important Tweets for News Writers using Recurrent Neural Network with Attention Mechanism and Multi-task Learning

Taro Miyazaki, Yuka Takei, Ichiro Yamada, Jun Goto
NHK Science & Technology Research Laboratories
1-10-11 Kinuta, Setagaya-ku, Tokyo, Japan
{miyazaki.t-jw, takei.y-ek,
yamada.i-hy, goto.j-fw}@nhk.or.jp

Shin Toriumi
Tokyo City University
3-3-2 Ushikubo-Nishi, Tsuzuki-ku
Yokohama, Kanagawa, Japan
g1683108@tcu.ac.jp

Abstract

Social media is an important source for news writers. However, extracting useful information for news writers from the vast amount of social media information is laborious. Therefore, services that enable news writers to extract important information are desired. In this paper, we describe a method to extract tweets that include useful information for news writers. Our method uses a Recurrent Neural Network (RNN) with an attention mechanism and multi-task learning that processes each character in the tweet to estimate whether the tweet includes important information. In our experiment, we compared two types of attention mechanism and compared their types with/without multi-task learning. By our proposed method, we obtained an F-measure of 0.627, which is 0.037 higher than that of baseline method.

1 Introduction

Social media information is now an important source for news writers. People who encounter an incident can post what is happening before his/her eyes using photos and videos. These posts are important primary information, so news writers want to gather them. However, extracting useful information from the vast amount of social media information is laborious. For this reason, services that enable news writers to extract information that can be used as a news source are desired. In fact, some services such as Spectee[1] and FASTALERT[2] have been launched in Japan. These services gather much information from social media and extract information that can be used as news sources.

Information that news writers want to extract from social media includes many different topics such as fires, accidents, and other incidents. Therefore, extracting information from social media by filtering with keywords is difficult. Assuming the words "delay" and "train" are included in the keywords, the tweet "xxx line is delayed by accident," which can be used as a news source, can be extracted. However, the tweet "I hope the train is delayed because I haven't studied for today's exam," which cannot be used as a news source, is also extracted. To extract tweets that include important information, filtering by keyword is not enough because the output may include tweets that cannot be used as news sources.

For this reason, we have been studying automatic extraction of useful information from social media. Our purposes are to reduce the amount of laborious work and extract information that cannot be extracted by using queries. In this paper, we describe a method to extract tweets that include useful information for news writers. Generally, social media posts are often written in colloquial style and often include abbreviations, slang and emojis. This makes word segmentation difficult. Therefore, our method is character-based approach, not a word-based one. Our method analyzes each character in a tweet by using a Recurrent Neural Network (RNN) and then decides whether the tweet includes important information. We adopted an attention mechanism and multi-task learning in our method and confirmed the effectiveness of our method. Our contribution is to

[1] http://www.spectee.com

[2] https://fa.xwire.jp

31st Pacific Asia Conference on Language, Information and Computation (PACLIC 31), pages 363–369
Cebu City, Philippines, November 16-18, 2017

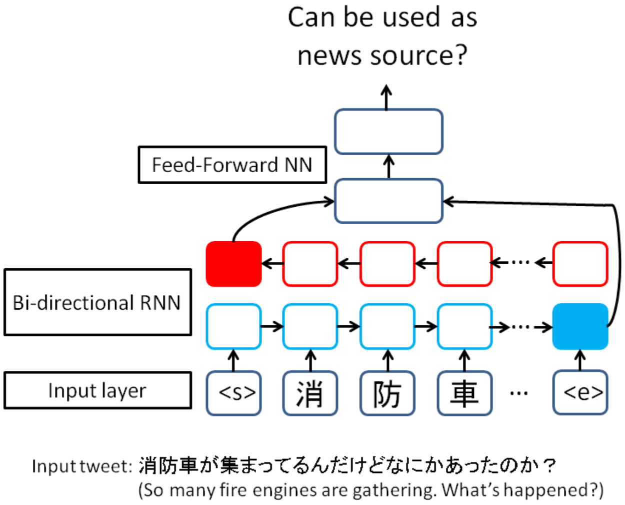
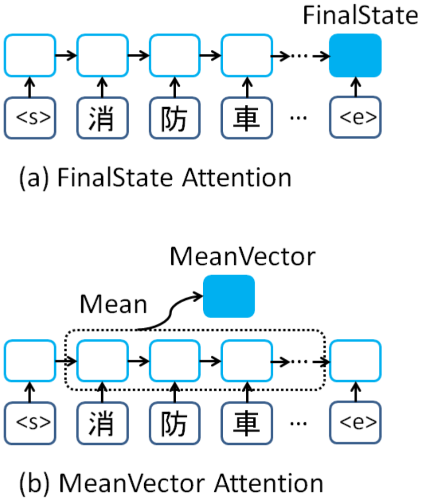

Can be used as news source?
Feed-Forward NN
Bi-directional RNN
Input layer
<s>
消
防
車
…
<e>
Input tweet: 消防車が集まってるんだけどなにかあったのか？
(So many fire engines are gathering. What's happened?)

Figure 1: Architecture of our basic system.



(a) FinalState Attention

(b) MeanVector Attention

Figure 2: Two types of attention.

reveal that the combination of attention mechanism and multi-task learning is effective for character-based approaches.

2 Methods for extracting important tweets

In this section, we describe our method for extracting important tweets for news writers. We use a Recurrent Neural Network (RNN)-based model as the basic method. And, we add the attention mechanism and multi-task learning.

2.1 Basic model (RNN-based model)

As we mentioned, sentences in social media are often written in colloquial style and often include abbreviations, slang and emojis. This makes it difficult to use state-of-the-art word segmentation or other natural language processing (NLP) tools. Japanese, our target language, is written without word separation, so the accuracy of the word segmentation directly affects the performance of word-based NLP tasks. Actually, according to Ling et al. (2015) and Dhingra et al. (2016), the character-based approach outperformed the word-based one in the social media analysis task. For these reasons, we chose characters, not words, as the input of our models.

Our basic method uses bi-directional RNN (biRNN) for obtaining vector representations of the input tweet. Each character in a tweet is sequentially inputted for both the forward and backward directions. When all characters are inputted, the final hidden states of the biRNN are used as vector representations. Then, our method classifies the tweet according to whether it is important by using a two-layer Feed-Forward Neural Network (FFNN). Figure 1 shows the architecture of our basic model.

2.2 Attention-based model

The attention mechanism has been used in many NLP tasks, such as machine translation (Bahdanau et al., 2015; Luong et al., 2015a) and image captioning (Xu et al., 2015), and can give weights to each input data taking into account the importance. In this paper, for comparison, we prepare two types of attention mechanism: "FinalState" and "MeanVector."

2.2.1 FinalState attention

FinalState attention is the conventional method. In this method, we calculate the attention weight using the final hidden state of biRNN (Figure 2-(a)). Here, we explain using the example of forward RNN: actually, we use both forward and backward RNNs.

The score for the t-th character $score_t$ is calculated as follows:

$$score_t = h_f^T \bar{h}_t$$

Here $\bar{h}_t$ is the hidden state of RNN, in which the t-th character of the input tweet has been inputted, and h_f is the hidden state of RNN, in which the final character of the tweet has been processed.

By using the score, the weight for the t-th character W_t is as follows:

$$W_t = \frac{\exp(score_t)}{\sum_{t'} \exp(score_{t'})}$$

Here, t' means the set of all characters in the tweet.

By using the weight and hidden state of the character, the FinalState attention a_f can be given:

$$a_f = \sum_{t'} W_t h_t$$

Our method uses the sum of a_f and h_f as a feature, and judges whether the tweet can be used as a news sources. The architecture of the FFNN consists of two layers as shown in Figure 1.

This method can give high weight to the characters that strongly affect the vector representation of the whole tweet because the higher the similarity between the t-th character's vector h_t and the tweet's vector h_f, the higher the weight W_t is.

2.2.2 MeanVector attention

MeanVector attention is our proposed method. In this method, we calculate the attention weight using the mean vector of the hidden state of biRNN for every character in the tweet (Figure 2-(b)). Similar to section 2.2.1, we explain using the example of the forward RNN.

The score for the t-th character $score_t$ is calculated as follows:

$$score_t = h_m^T \bar{h}_t$$

$$h_m = \frac{\sum_{t'} \bar{h}_t}{t'}$$

By using $score_t$, we can calculate the weight W_t and MeanVector attention a_m in a similar way to that in section 2.2.1.

$$W_t = \frac{\exp(score_t)}{\sum_{t'} \exp(score_{t'})}$$

$$a_m = \sum_{t'} W_t h_t$$

We use the sum of a_m and h_f as the input of the two-layer FFNN, and judge the tweet according to whether it can be used as a news source.

MeanVector attention can also give high weight to the characters that strongly affect the meaning of the whole tweet. However, compared with FinalState attention, the effect of the position of the character appeared to be reduced.

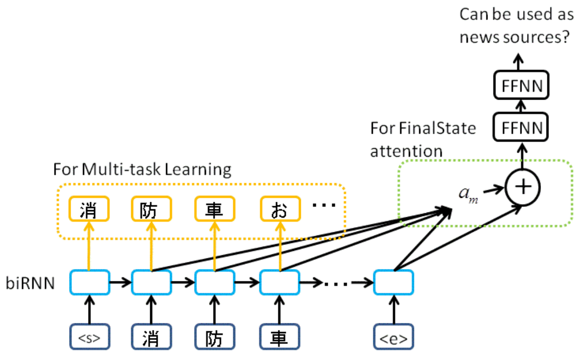


Figure 3: Overall structure using FinalState attention and multi-task learning.

2.3 Multi-task learning

In some studies using a neural network, models are trained with multiple tasks. This technique is called "multi-task learning." It has been reported that by using multi-task learning, the model can be generic and accurate (Luong et al., 2015b; Søgaard et al., 2016). Therefore we also use multi-task learning so that our model is more accurate.

In addition to our target task, by judging whether the tweet can be used as a news sources, we prepare the task that involves estimating the next character of the input character as another task. This task is the same as "neural language model learning." We do not need to prepare training data for this task; we can use the same dataset as that of the target task without new annotated data.

We designed our architecture as sharing input and a biRNN layer with these two tasks and prepared two output layers for each task. To train this model, first we start to train with the neural language model learning. After finishing this task, we start to train for the target task using the results of the first training as the initial model of the target task.

Figure 3 illustrates the overall structure of our method using FinalState attention and multi-task learning.

3 Experiment

3.1 Dataset

For training data, we gathered tweets that can be used as actual news sources as positive samples and randomly sampled tweets as negative samples. NHK's social listening team gathered positive samples. About 20 people work for the team. The team members have been working every day for about three years as professionals, so they are well trained and highly reliable. Negative samples were randomly sampled, so some positive samples are included in the data. However, there are not that many positive samples[3] , so we regarded that their effect is limited.

For evaluation data, we prepared two datasets. One was gathered in a similar way to that of gathering the training data. The ratio of positive to negative samples are adjusted to be almost the same as that of the actual tweets to reflect actual usage of our method in the news reporting section. This dataset is named "all the data" in this paper.

The other was gathered from 2,000 tweets by using a combination of queries, which include about 180 words connected with "and" / "or" / "not"[4]. These tweets are annotated according to whether they can be used as news sources by one evaluator. This dataset is designed to consider actual use by a news reporting section of a broadcasting company. Compared to the task of the other dataset, this is a more difficult task because all the data in this dataset includes some news-related words. This data is named "filtered data" in this paper.

The size of each dataset is given in Table 1. The hashtags, user names, HTML tags, and URLs are removed from all tweets in the datasets. The ratio of positive / negative samples in the training data does not reflect real-world distribution because of the training cost.

3.2 Implementation

We use Chainer (Tokui et al., 2015) to implement our models.

Table 1: Size of each dataset

Dataset		Amount
Training data	Positive sample	19,962
	Negative sample	1,524,155
All the data	Positive sample	2,582
	Negative sample	190,011
Filtered data	Positive sample	426
	Negative sample	1,574

Table 2: Results of evaluation using all the data

Method	Recall	Precision	F-measure
Without attention	0.887	0.872	0.880
+ multi-task	0.945	0.765	0.846
FinalState	0.861	0.922	0.890
+ multi-task	0.860	0.932	0.894
MeanVector	0.912	0.871	0.891
+ multi-task	0.906	0.875	0.890
Query filtering	0.495	0.893	0.637

For the middle layer, we use Long Short-Term Memory (LSTM) for each model of biRNN with 200 hidden states, and FFNN with a unit size of 200 and 100 from the near side of the input layer. The number of epochs is set to 10 for the target task and 3 for the neural language model learning (for multi-task learning). The mini-batch size is set to 200. We use Adam (Kingma and Ba, 2014) to optimize the parameters and Exponential Linear Units (ELUs) (Clevert et al., 2015) to activate function.

3.3 Evaluation results

3.3.1 Evaluation using all the data

The results of the evaluation experiment using all the data are given in Table 2. "Query filtering" uses the combination of queries mentioned in section 3.1 and is shown for reference.

This table indicates that all methods outperformed query filtering, but the differences between each method are small. Therefore, we try another evaluation to find out the differences.

3.3.2 Evaluation using filtered data

The results of the evaluation experiment using filtered data are given in Table 3. The accuracy of each method is lower than that shown in Table 2 because this task is far more difficult than the task in the other

[3] We gathered 8 million tweets every day, and the number of positive samples is less than 8,000. Therefore, the positive / negative rate is about 0.1%.

[4] These queries are used by a social media analysis team in a broadcast company in Japan.

Table 3: Results of evaluation using filtered data

Method	Recall	Precision	F-measure
Without attention	0.615	0.567	0.590
+ multi-task	0.552	0.622	0.585
FinalState	0.580	0.572	0.576
+ multi-task	0.674	0.573	0.619
MeanVector	0.650	0.535	0.587
+ multi-task	0.650	0.606	0.627
Query filtering	1.000	0.213	0.351

experiment.

In total, the method using MeanVector attention with multi-task learning, which is our proposed method, obtained the highest F-measure, which is 0.037 higher than that of the basic method without attention and multi-task learning.

The result of the MeanVector attention method is rather good compared to those of the two attention mechanisms. Both attention methods cannot increase the accuracy from basic method without multi-task learning, but using multi-task learning increased the F-measures of both attention methods.

We can say that MeanVector attention is better, and multi-task learning is necessary for this task.

3.4 Discussion

MeanVector attention outperforms FinalState attention. Figure 4 shows the comparison of the attention weight for each character between the two attention methods. The input tweet is “For some reason, policemen are gathering in front of my house, haha.” This tweet can be a news source because “policemen are gathering” may mean an incident has occurred near the place where the post was written. In the example in Figure 4, the FinalState attention method gives high weight to the character “笑 ” (haha). In this way, the FinalState attention method tends to give high weight to the characters that appear at the end of the tweet. On the other hand, MeanVector attention gives weight without being affected much by the position of the character in the tweet. Therefore, MeanVector attention gives high weight to “警察” (policemen) and “集まる” (gathering). This is the reason MeanVector attention outperforms the other attention method in this task.

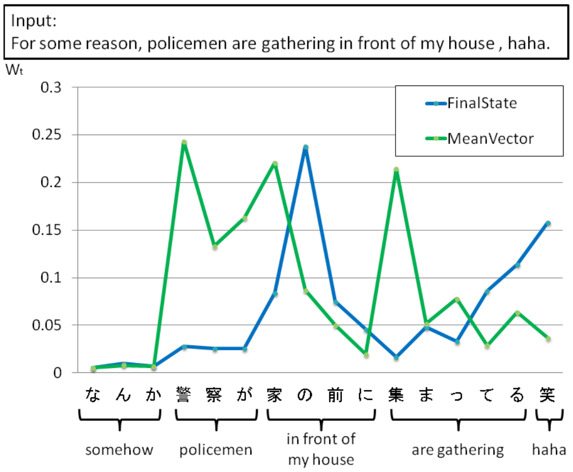


Figure 4: Comparing two attention methods.

In our experiment, which is described in section 3.3.2, the methods with attention mechanism without multi-task learning did not increase the accuracy; however, using multi-task learning increased the accuracy. Calculating the attention weight uses the hidden states of biRNN for each input character. We assume that the methods using attention without multi-task learning cannot learn the hidden state precisely, and multi-task learning allows the models to learn the hidden state for each input character more precisely. For more details, losses for the model update are calculated only once per mini-batch in our “target task” training. This means that only one loss covers the RNN of each input character. On the other hand, in our “neural language model” training, losses are calculated for each input character, and these losses are much more than those of “target task” training. Therefore “neural language model” training can help the model to learn more precisely for each input character. By using the “neural language model” trained model as the initial model of the “target task,” we can obtain the hidden states of biRNN that are used for calculating attention more precisely. As a result, the attention weight for each character can be calculated more precisely by using multi-task learning, so the attention mechanism works well. Figure 5 shows the effect of using multi-task learning from the viewpoint of the attention weight for each character. The input tweet is “Awful, Sakai-gawa river may overflow.” This can be used as a news source because we can find out from the tweet that a river is now dangerous. The method without multi-task learning gives almost constant at-

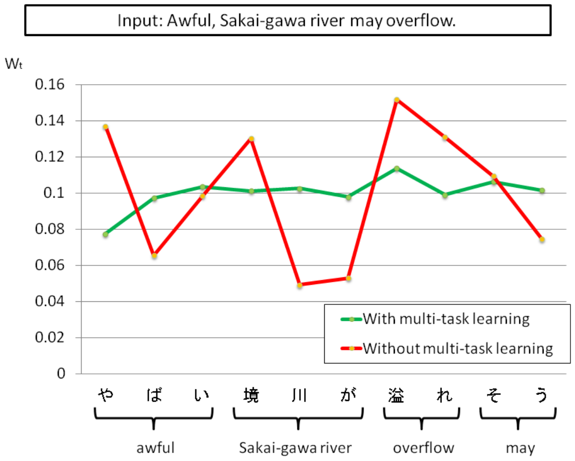


Figure 5: Effect of using multi-task learning.

tention weight. On the other hand, the method with multi-task learning gives high attention weight to "溢れ" (overflow). For this reason, using multi-task learning made the accuracy high.

The ratio of positive / negative samples in our training data is quite imbalanced, and our methods did not includes any special features to manage the imbalanced. However our methods can perform with rather high accuracy. Therefore, we can say that our neural network can overcome the imbalanced training data.

4 Related work

Large-scale social media analysis systems named "DISAANA," and "D-SUMM" are now in operation (Mizuno et al., 2016). These systems analyze tweets as information sources and extract useful information to assess the damage caused by large-scale disasters.

There are many studies on extracting information that can be used as a news source (Vosecky et al., 2013; Hayashi et al., 2015). They used the bag-of-words approach and obtained good results. However, as we mentioned, tweets are often written in colloquial style, so word segmentation is difficult. Moreover, generally, tweets include too many words to handle, so we avoid using bag-of-words approach.

To do this, there are some studies that use a character-based approach to handle tweets (Vosoughi et al., 2016; Dhingra et al., 2016; Vakulenko et al., 2017). By using these approaches, word segmentation is not necessary, and the vocabulary to be handled is reduced; however, there are few studies that use the attention mechanism in character-based approaches. We used a similar architecture to that of the Tweet2Vec model (Dhingra et al., 2016). We expanded this model by using the attention mechanism and multi-task learning.

We referred to some studies for our future work. Dredze et al. (2016) estimated the geolocation of the tweet. They used the time the tweet was written and obtained good results. Chi et al. (2016) used textual features selected based on a frequency-based feature selection strategy. Kanouchi st al. (2015) classified each tweet according to the people who was mentioned in the tweet, such as the person who posted the tweet of himself/herself, his/her family or people around him/her. These studies are useful to detect where and who is the subject of the tweet, which is important to news writers, who are our target users.

5 Conclusion

We presented a method to extract tweets that can be used as news sources using a recurrent neural network with attention and multi-task learning. In this paper, we confirmed the effect of the attention mechanism and multi-task learning in our task. Comparing the two methods of attention mechanism, FinalState and MeanVector, we showed that the MeanVector method is better in our task. Overall, our method (MeanVector attention with multi-task learning) achieved an F-measure of 0.627 in F-measure, which is 0.037 higher than baseline method. In our experiment, the attention mechanism is effective only when used with multi-task learning.

Our future work is adding new features according to the task, multi-class classifying to detect which kind of incident is mentioned in the tweet, and extracting more information such as geolocation and the subject person of the tweet.

References

Dzmitry Bahdanau, Kyunghun Cho and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translation. *arXiv: 1409.0473*.

Lianhua Chi, Kwan Hui Lim, Nebula Alam and Christopher K. Butler. 2016. Geolocation Prediction in Twitter Using Location Indicative Words and Textual Features. *In proceedings of the 2nd Workshop on Noisy User-generated Text: 227–234.*

Djork-Arné Clevert, Thomas Unterthiner and Sepp Hochreiter. 2015. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). *arXiv: 1511.07289.*

Bhuwan Dhingra, Zhong Zhou, Dylan Fitzpatrick, Michael Muehl and William W. Cohen. 2016. Tweet2Vec: Character-Based Distributed Representations for Social Media. *In Proceeding of the 54th Annual Meeting of the Association for Computational Linguistics (ACL): 269–274.*

Mark Dredze, Miless Osborne and Prabhanjan Kambadur. 2016. Geolocation for Twitter: Timing Matters. *In proceedings of NAACL-HLT: 1064–1069.*

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 21st International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD): 417–426*

Shin Kanouchi, Mamoru Komachi, Naoaki Okazaki, Eiji Aramaki and Hiroshi Ishikawa 2015. Who caught a clod? – Identifying the subject of a symptom. *In proceedings of the 53rd Annual Meeting of the Association for Computational Linguistic (ACL): 1660-1670.*

Diederik Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv: 1412.6980.*

Wang Ling, Tiago Luís, Luís Marujo, Ramón F. Astudillo, Silvio Amir, Chris Dyer, Alan W. Black and Isabel Trancoso. 2015. Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation. *arXiv: 1508.02096.*

Minh-Thang Luong, Hieu Pham and Christopher D. Manning. 2015a. Effective Approaches to Attention-based Neural Machine Translation. *arXiv: 1508.04025.*

Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals and Lukasz Kaiser. 2015b. Multi-task Sequence to Sequence Learning. *arXiv: 1511.06114.*

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 the 26th International Conference on Computational Linguistics (COLING): 263–267*

Seiya Tokui, Kenta Oono, Shohei Hido and Justin Clayton. 2015. Chainer: a Next-generation Open Source Framework for Deep Learning. *In proceedings of NIPS15 Worksyop on LearningSys.*

Anders Søgaard and Yoav Goldberg. 2016. Deep Multitask Learning with Low Level Tasks Supervised at Lower Layers. *In proceedings of 54th Annual Meeting of the Association for Computational Linguistics (ACL): 231-235.*

Svitlana Vakulenko, Lyndon Nixon and Mihai Lupo. 2017. Character-based Neural Embedding for Tweet Clustering. *In proceedings of the Fifth International Workshop on Natural Language Processing for Social Media: 36-44.*

Soroush Vosoughi, Prashanth Vijayaraghavan and Deb Roy 2016. Tweet2Vec: Learning Tweet Embeddings using Character-level CNN-LSTM Encoder-Decoder. *In proceedings of the 39th International conference on Research and Development in Information Retrieval (ACM SIGIR): 1041–1044.*

Jan Vosecky, Di Jiang, Kenneth Wai-Ting Leung and Wilfred Ng. 2013. Dynamic Multi-Faceted Topic Discovery in Twitter. *In proceedings of International Conference on Information and Knowledge Management (CIKM): 879–884.*

Kelvin Xu, Jimmy Lei Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel and Yoshua Bengio. 2015. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. *In proceedings of the 32nd International Conference on Machine Learning (ICML): 77–81.*