NTCIR13 MedWeb Task: Multi-label Classification of Tweets using an Ensemble of Neural Networks

Hayate Iso
Nara Institute of Science and Technology, JAPAN
iso.hayate.id3@is.naist.jp

Camille Ruiz
Nara Institute of Science and Technology, JAPAN
camille.ruiz.co3@is.naist.jp

Taichi Murayama
Nara Institute of Science and Technology, JAPAN
murayama.taichi.mk1@is.naist.jp

Katsuya Taguchi
Nara Institute of Science and Technology, JAPAN
taguchi.katsuya.tb3@is.naist.jp

Ryo Takeuchi
Nara Institute of Science and Technology, JAPAN
takeuchi.ryo.tj7@is.naist.jp

Hideya Yamamoto
Nara Institute of Science and Technology, JAPAN
yamamoto.hideya.xx7@is.naist.jp

Shoko Wakamiya
Nara Institute of Science and Technology, JAPAN
wakamiya@is.naist.jp

Eiji Aramaki
Nara Institute of Science and Technology, JAPAN
aramaki@is.naist.jp

ABSTRACT

This paper describes how we tackled the Medical Natural Language Processing for Web Document (MedWeb) task as participants of NTCIR13. We utilized multi-language learning to integrate the multi-language inputs of the task into a single neural network. We then built two neural networks–a hierarchical attention network (HAN) and a deep character convolutional neural network (CharCNN)–with multi-language learning and combined both outputs to utilize the advantages of each neural network. This combination was carried out using ensembling, specifically the method of bagging. We found that the ensemble using the loss functions NLL and hinge produced the best results with 88.0% accuracy.

Team Name

NAIST

Subtasks

MedWeb (Japanese, English, Chinese)

Keywords

MedWeb, NTCIR13, Sentence Classification, Multi-label, Disease related tweets.

1. INTRODUCTION

The NTCIR13 Medical Natural Language Processing for Web Document (MedWeb) task challenges participants to classify Twitter-like messages written in different languages (namely Japanese, English, and Chinese). In particular, the participants were asked to perform a multi-label classification of 8 diseases or symptoms for each tweet. Further details of this task can be found in the NTCIR MedWeb website and the NTCIR-13 Medweb task overview paper [11].

It was given that the English and Chinese datasets were generated from the Japanese dataset by translating Japanese sentences to these languages. In light of this data generation, we found that all of the language datasets have the same label for corresponding sentences.

From this observation, we propose a neural network that could handle multi-language inputs with the same labels. Further, we explore ensemble strategies that are suitable for the MedWeb task.

Our results were then compared to the 2nd place teams for each language, AKBL-ja-3, UE-en-1, and TUA1-zh-3–revealing that most of our experiments are superior to the benchmarks' result. Using exact match, we found that the ensemble of the attention network and the character-level convolutional network using the negative log likelihood (NLL) and hinge loss functions produced the best results with 88.0% accuracy.

2. METHOD

Our modeling strategies are summarized in Figure 1. Our team tackled the MedWeb task using a neural network classifier.

We first introduce two neural network models for text classification described in Section 2.1. We then outline our multi-label strategy in Section 2.2 We show the learning strategies for each neural model in Section 2.3 especially since the selection of the loss function had a significant effect on model robustness. Finally, we present the ensemble method called bagging in Section 2.4.

2.1 Tweet Encoding Methods

In this paper, we utilized two neural network models based on both Hierarchical Attention Network (HAN) [12] and Character-level Convolutional Networks (CharCNN) [13]. The goal is to encode the tweet sentence into a fixed size sentence vector s, which will eventually undergo multi-label classification.

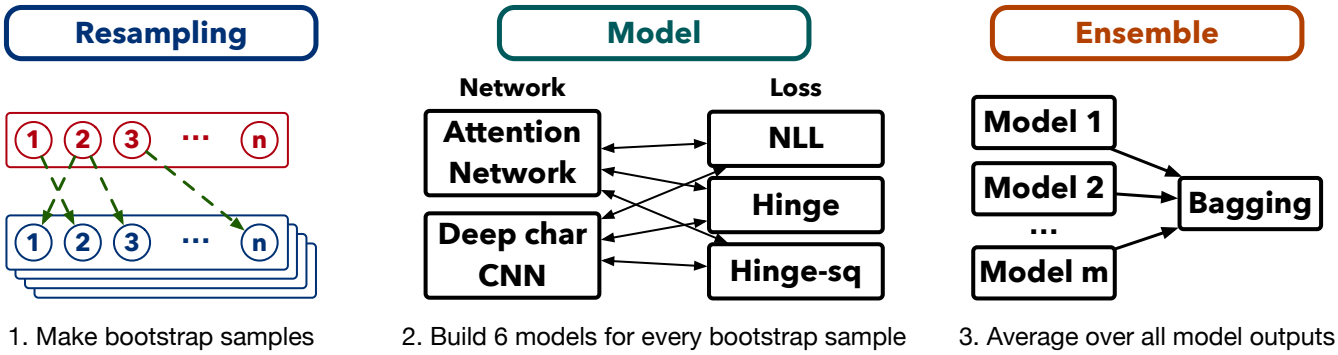
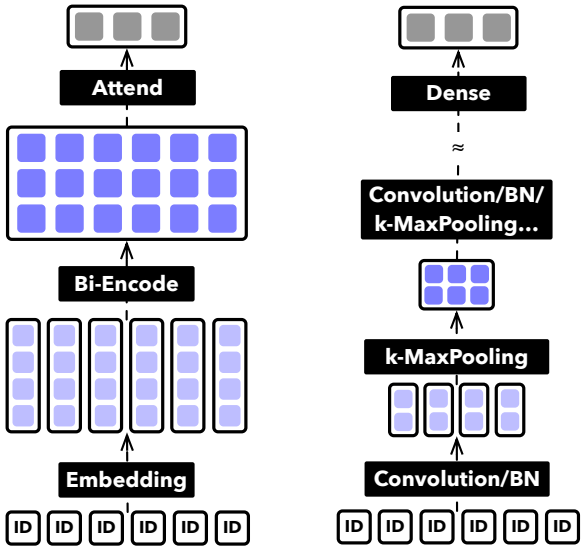


Figure 1: Overview: To make a more robust classifier, we created 20 bootstrap dataset samples, instead of a single sample, to be fed to our models. We then generated 6 methods (with respect to two neural networks and three loss functions) for each bootstrap sample–which amounts up as 120 model combinations. Finally, we merged the 120 outputs of these models into a single result by taking the average of these outputs.



(a) One step Attention Network

(b) Character-level convolutional neural network

Figure 2: This figure represents the neural network models we used for this task. The first is a simplified variation of the hierarchical attention network while the second is a variation of the convolutional neural network.

2.1.1 *One Step Hierarchical Attention Network*

Our approach includes using a one step hierarchical attention neural network classifier (see Figure 2a). An attention network processes data sequentially rather than all at once. The original model, HAN, assumes the hierarchical structure from word to sentence to document. However, for the dataset of this task, there is only one step that is relevant–word to tweet (sentence). For this reason, we use a one step hierarchical attention network.

For HAN, we first have to embed the word of a tweet. In a tweet, we are given a sentence with words w_t where T is the total number of words in the sentence. We embed these words through the embedding matrix $W_e, x_t = W_e w_t$ and perform a bidirectional GRU [5]. The forward GRU $\overrightarrow{f}$ reads the words in the sentence from the first word w_1 to the last word w_T while the backward GRU $\overleftarrow{f}$ reads the words in the sentence from last word w_T to first word w_1.

$$x_t = W_e w_t, t \in [1, T]$$
$$\overrightarrow{h}_t = \overrightarrow{\text{GRU}}\left(x_t\right), t \in [1, T],$$
$$\overleftarrow{h}_t = \overleftarrow{\text{GRU}}\left(x_t\right), t \in [T, 1],$$

We then concatenate $\overrightarrow{h}_t$ and $\overleftarrow{h}_t$ to get h_t which serves as the annotation for the word w_t. This means that h_t is a summary of the information of the tweet with respect to the word w_t.

Next, we employ an attention mechanism [1] to represent the sentence vector s through combining h_t:

$$u_t = \tanh\left(W_w h_t + b_w\right)$$
$$\alpha_t = \frac{\exp(u_t^\top u_w)}{\sum_t \exp(u_t^\top u_w)}$$
$$s = \sum_t \alpha_t h_t$$

2.1.2 *Character-level Convolutional Network with Batch Normalization*

Another sentence encoder that we used is the character-level convolutional network (CharCNN) based on the Zhang et al's model[13] (see Figure 2b).

In contrast to the word embedding from HAN, CharCNN focuses more on characters than words. Given a tweet, there is a one-hot vector of character c_t representing the tth character of a tweet with length T_c. If we let the concatenation operator be $\oplus$, then a sentence of length T_c can be represented as

$$c_{1:n} = c_1 \oplus c_2 \oplus \cdots \oplus c_{T_c}.$$

In general, let $c_{i:j}$ refer to the concatenation of characters $c_i \oplus c_{i+1} \oplus \cdots \oplus c_j$. A convolution is defined as a filter W_c with window size r to produce hidden features z_1:

$$z_{i,1} = \text{Relu}(W_{c,1} c_{i:i+r} + b_1).$$

To accelerate the learning procedure, we apply batch normalization (BN) [6] for hidden features z_1:

$$\hat{z}_{i,1} = \gamma_1 \left(\frac{z_{i,1} - E_{\mathcal{B}}[z_{i,1}]}{\sqrt{V_{\mathcal{B}}[z_{i,1}]}} \right) + \beta.$$

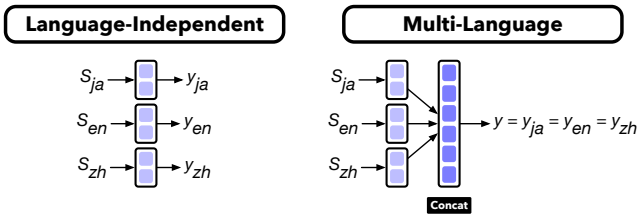


Figure 3: This depicts the difference between language-independent learning and multi-language learning.

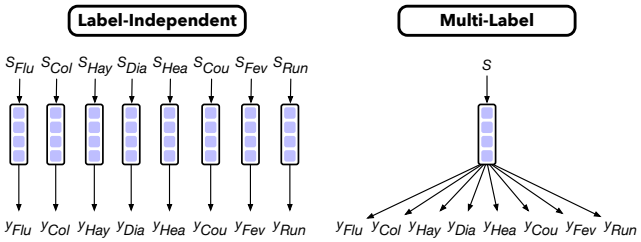


Figure 4: This shows the difference between label-independent learning and multi-label learning.

where $\mathcal{B}$ indicates the set of mini-batch instances.

A max-pooling layer of size k is then applied to extract the valuable features from hidden features $\hat{z}_{1,1:T_c-r+1}$:

$$v_{1,j} = \max\left(\hat{z}_{1,j:j+k}\right)$$

for every $j \in \{k \times i : i \leq \lfloor T_c - r + 1/k \rfloor\}$.

We define the above procedure as a convolution operation denoted by $\textsc{Cnn}(\cdot)$. We iterate the convolution operation three times:

$$\begin{aligned} v_{1,1:T_{v,1}} &= \textsc{Cnn}(c_{1:T_c}) \\ v_{2,1:T_{v,2}} &= \textsc{Cnn}(v_{1,1:T_{v,1}}) \\ v_{3,1:T_{v,3}} &= \textsc{Cnn}(v_{2,1:T_{v,2}}) \end{aligned}$$

where $T_{v,1} = \lfloor T_c/k \rfloor, T_{v,2} = \lfloor T_{v,1}/k \rfloor, T_{v,3} = \lfloor T_{v,2}/k \rfloor$.

We then apply the linear transformation for hidden features v_3 to compose the sentence vector s:

$$s = W_v v_3 + b_v.$$

2.1.3 Multi-Language Learning

As mentioned in Section 1, our approach integrates multi-language input into a single neural network. Although we generally need to learn the neural network model for each task, the MedWeb task consists of the same label set for the different language datasets. This results in a simultaneous learning of the Japanese, English, and Chinese dataset (see Figure 3). Multi-language inputs are able to use richer information of sentences as it combines these three languages.

A general learning strategy that we call Language independent learning involves learning each tweet per language. We compare this strategy against multi-language learning. Multi-language learning involves learning using multiple languages of the same label. Since the English and Chinese tweets are translated from original Japanese tweets, all languages use the same tweet and the same label per line. Specifically, the Japanese, English, and Chinese tweets in the n-th line of the dataset have the same label. In light of this observation, we propose that multi-language learning is more suitable for this task.

Since we found that corresponding tweets represent the same content but vary in language, a single sentence vector can represent these tweets. Thus, we represent the three tweets of each language in a single vector for multi-language learning:

$$s^{\text{Multi}} = [s^{\text{ja}}; s^{\text{en}}; s^{\text{zh}}]$$

where $[\cdot;\cdot]$ represents vector concatenation.

2.2 Multi-Label Learning

Since the task is to perform a multi-label classification of 8 diseases or symptoms per tweet, there are two ways to approach this (see Figure 4). The first is to build the classifier for each label, respectively, which we call *Label-independent learning*.

$$\hat{y}_c = w_c^\top s + b'_c \in \mathbb{R}$$

Another approach is to classify the tweet for the 8 labels simultaneously. This is called multi-label learning [10]:

$$\hat{y} = W_c s + b_c \in \mathbb{R}^8$$

The dimension of outputs $\hat{y}$ equals 8 because each sentence has 8 binary labels: *Influenza, Cold, Hay Fever, Diarrhea, Headache, Cough, Fever* and *Runny nose*.

The advantages of multi-label learning is that the model could share the same sentence vector s of tweets for all labels. In the multi-label setting, the sentence vector s could capture the symptom co-occurrence pattens.

To determine the predicted labels for each symptom c, we apply the sign function for each output as follows:

$$\text{sgn}(\hat{y}_c) = \begin{cases} 1 & \text{if } \hat{y}_c \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

2.3 Optimization Methods with Different Loss Function

To optimize the models that we introduced, several approaches are applicable. The each model's robustness is highly influenced by the selection of loss function [8]. Although negative log likelihood (NLL) is generally applied to optimize a classifier, we further experimented with other loss functions–specifically hinge and squared hinge:

$$\mathcal{L}_{\text{NLL}} = \sum_i^N \sum_{c=1}^8 \ln(1 + \exp(-y_{c,i}\hat{y}_{c,i}))$$

$$\mathcal{L}_{\text{Hinge}} = \sum_i^N \sum_{c=1}^8 \max(0, 1 - y_{c,i}\hat{y}_{c,i})$$

$$\mathcal{L}_{\text{Hinge-sq}} = \sum_i^N \sum_{c=1}^8 \max(0, 1 - y_{c,i}\hat{y}_{c,i})^2$$

where $y_{c,i} \in \{-1, +1\}$ is the true label for tweet i's symptom c, N is the number of training instances and 8 indicates the number of symptoms.

2.4 Ensembles Using Bagging

The ensemble is a conventional strategy of improving classification accuracy. Instead of using and improving one classifier for the NTCIR MedWeb task, we took the ensembles of

Table 1: Data summary

	Avg #word		Max #word	Vocabulary of word	Avg #char		Max #char	Vocabulary of char
	Train	Test			Train	Test		
ja	13.6	12.5	53	2161	35.1	33.6	142	958
en	13.2	11.9	60	1819	68.0	63.4	282	45
zh	11.4	10.0	41	2298	27.9	25.9	101	1236

strong machine learning methods. For this task, we focused on neural networks–namely Hierarchical Attention Neural Network (HAN) and Character-level Convolutional Neural Network (CharCNN). To get the best results, we sought to improve these classifiers by experimenting with various combinations of loss functions.

Ensembling is a machine learning algorithm that combines multiple methods to boost classification accuracy. Normally, since these tend to have low accuracies for complicated tasks, weak machine learning algorithms are combined into an ensemble, not strong ones. However, in this experiment, we create ensembles of strong machine learning methods–namely HAN and CharCNN. For this experiment, we used bagging to ensemble these methods [3]. Bagging first creates N' datasets by resampling the original dataset. An average is then computed over the model outputs from each resampled dataset:

$$\hat{y}^{Bagging} = \frac{1}{|N'| \cdot |M|} \sum_{n \in N'} \sum_{m \in M} \hat{y}_{n,m}$$

where N' is the set of resampled datasets, M is the set of the model architectures and each $\hat{y}_{n,m}$ is an output of the model m trained on the n-th resampled dataset.

We made 20 resampled datasets for this purpose. We use each dataset for training the HAN and CharCNN against the 3 loss functions, resulting in 6 methods. As a result of this procedure, these 6 methods are subjected to each resampled dataset which generates a total of 120 models to be learned. We tested and compared various ensembles of these models to identify the best classifiers for this task.

3. MATERIAL

The statistics of the MedWeb dataset are summarized in Table 1. The English and Chinese datasets are generated from translating sentences from the Japanese datasets-making the set of their labels identical.

In order to tokenize each of the sentences, we used the NLTK TweetTokenizer [2], MeCab [7], and jieba [9] for our English, Japanese, and Chinese tokenizers respectively.

4. RESULTS AND DISCUSSION

In this section, we discuss the results of our experiments. All of the models are implemented with Keras [4]. Our model, ultimately, obtained an 88.0% exact match accuracy.

4.1 Multi-label learning effects

For comparing label independent learning and multi-label learning, we classified Japanese tweets using the HAN classifier and NLL loss function. The corresponding accuracies are shown side by side in Table 2. While there is limited improvement from label independent learning to multi-label learning when we observe each label, the exact match accuracy significantly improves from 76.7% to 82.3%.

Table 2: Comparison between label-independent or multi-label

Target	Exact match accuracy	
	Label-Independent	Multi-Label
Influenza	0.977	**0.988**
Diarrhea	0.973	**0.979**
Hay Fever	0.971	**0.975**
Cough	0.988	**0.991**
Headache	0.979	**0.981**
Fever	**0.931**	0.929
Runny nose	0.948	**0.952**
Cold	0.944	**0.965**
Exact match	0.767	**0.823**

We suspect that the results are as such because learning via multi-label learning is able to take into account the connections between diseases. For example, people with influenza may also have colds or fever. On the other hand, learning labels independently may not be taking this into account. Hence, multi-label learning generally gives better results compared to independent label learning.

Since multi-label learning is generally a better method for this task, we used multi-label learning in our experiment first with language learning and then with ensembles.

4.2 Multi-Language Learning and Bagging

After comparing multi-language learning to language independent learning, the results show that multi-language learning with bagging produces good results for all combinations of classifiers and loss functions (see Table 3). Our results also show that multi-language learning with bagging is also better than multi-language learning on its own–depicting how reducing generalization error through bagging produces better results.

Nonetheless, our results for language independent learning already surpass the results of the benchmark (Table 3): 82.5% for Japanese (AKBL-ja-3: 80.5%), 79.5% for English (UE-en-1: 78.9%), and 80.9% for Chinese (TUA1-zh-3: 78.6%). We further exceed these label independent results through single multi-label learning, or multi-label learning without bagging, where our highest score is at 84.4% with HAN and hinge. This shows that the multi-label learning in general is better than language-independent learning for this task. Finally, our score further improves with multi-label learning with bagging: 86.9% accuracy using Character level CNN with hinge loss function.

This segment of the experiment shows that it is better to use multi-language learning with bagging for our dataset. We then used multi-language learning with bagging when we classify tweets with ensembles of neural networks.

Table 3: Language Independent Learning vs. Multi-language Learning - This table shows that multi-language learning is more accurate than language independent learning in any of the languages and classifiers for this dataset. We also append the other team's results for each language, AKBL-ja-3, UE-en-1, TUA1-zh-3 for benchmark, respectively.

Setting		Exact match accuracy				
Encode	Loss	Language-Independent			Multi-Language	
		ja	en	zh	Single	Ensemble
Attention	NLL	0.823	0.791	0.789	0.823	0.841
	Hinge	0.823	**0.795**	**0.809**	**0.844**	0.841
	Hinge-sq	**0.825**	0.786	0.794	0.822	0.844
CharCNN	NLL	0.800	0.718	0.808	0.831	0.848
	Hinge	0.797	0.686	0.806	0.811	**0.869**
	Hinge-sq	0.772	0.670	0.784	0.811	0.866
Benchmark		0.805	0.789	0.786	-	-

Table 4: This table shows the results of our ensembles. Among the 9 ensembles we created, we submitted the last 3–particularly the ensembles using both HAN and CharCNN. Of the three, the ensemble with loss functions NLL and Hinge produced the highest accuracy: 88.0%.

Submission Id	Ensemble strategy		Exact match accuracy
	Encode	Loss	
-	Attention	NLL × Hinge × Hinge-sq	0.842
-		NLL × Hinge	0.836
-		NLL × Hinge-sq	0.844
-	CNN	NLL × Hinge × Hinge-sq	0.861
-		NLL × Hinge	0.861
-		NLL × Hinge-sq	0.859
NAIST-*-1	Attention × CNN	NLL × Hinge × Hinge-sq	0.877
NAIST-*-2		NLL × Hinge	**0.880**
NAIST-*-3		NLL × Hinge-sq	0.878

4.3 Classification: Ensembles of Loss Functions

We have 9 variations of ensembles that are taken from the combinations of neural networks HAN, CNN, and HAN-CNN with loss functions NLL-hinge-hinge squared, NLL-hinge, and NLL-hinge squared (see Table 4). Of the 9, 3 were submitted for the MedWeb Task, namely (1) HAN and CNN with NLL and Hinge, (2) HAN and CNN with NLL and Hinge Squared, and (3) HAN and CNN with NLL, Hinge, and Hinge Squared.

Our results here are superior to our previous results. We garnered an 88.0% accuracy with an ensemble of HAN and CNN with NLL and Hinge (see Table 4). This shows how ensembling boosts the accuracy of the already strong machine learning methods of HAN and CNN.

We submitted the same three models for all language tasks because the label set is the same with respect to the language difference.

5. CONCLUSION

This paper discusses Team NAIST's submission to NTCIR 13 MedWeb task. The task was to classify whether or not pseudo-tweets refer to certain symptoms or diseases. The task requires a multi-label classification of tweets coming from a multi-lingual corpus (Japanese, English and Chinese) which can contain up to 8 diseases.

We experimented with multi-label and label-independent learning, where we found that multi-label learning provides better results for this task. Moreover, we also looked into multi-language and language independent learning, where we found that multi-language learning with bagging provides superior results. Finally, with multi-label and multi-language learning, the ensemble of hierarchical attention network (HAN) and deep character-level convolutional neural network (CNN) with loss functions negative log likelihood (NLL) and hinge provided the highest accuracy of 88.0%.

Acknowledgements

This work was supported by Japan Agency for Medical Research and Development (Grant Number: 16768699) and JST ACT-I.

6. REFERENCES

[1] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2015.

[2] S. Bird. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 69–72. Association for Computational Linguistics, 2006.

[3] L. Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[4] F. Chollet et al. Keras. https://github.com/fchollet/keras, 2015.

[5] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[6] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 448–456, 2015.

[7] T. Kudo, K. Yamamoto, and Y. Matsumoto. Applying conditional random fields to japanese morphological analysis. In *EMNLP*, volume 4, pages 230–237, 2004.

[8] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of machine learning*. MIT press, 2012.

[9] J. Sun. jieba. https://github.com/fxsjy/jieba, 2013.

[10] G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, 3(3), 2006.

[11] S. Wakamiya, M. Mizuki, K. Yoshinobu, O. Tomoko, and E. Aramaki. Overview of the ntcir-13: Medweb task. In *Proceeding of the NTCIR-13 Conference*, 2017.

[12] Z. Yang, D. Yang, C. Dyer, X. He, A. J. Smola, and E. H. Hovy. Hierarchical attention networks for document classification. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL)*, pages 1480–1489, 2016.

[13] X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. In *Proceedings of the Advances in neural information processing systems (NIPS)*, pages 649–657, 2015.