Weakly-Supervised Neural Categorization of Wikipedia articles

Xingyu Chen † Mizuho Iwaihara ‡

† Graduate School of Information, Production and System, Waseda University

Kitakyushu 808-0135 Japan

E-mail: † chenxingyu@akane.waseda.jp, ‡ iwaihara@waseda.jp

Abstract Wikipedia is the largest online encyclopedia, in which articles are organized into hierarchical categories. One category in Wikipedia consists of its member articles and subcategories. The task of Wikipedia article categorization is that given a new article, finding which category it should belong to. Deep neural models are gaining increasing popularity for many NLP task, due to their strong expressive power and less requirement for feature engineering. However, neural models always need a large amount of labeled training data, while one Wikipedia category does not have enough member articles. In this paper, we utilize a weakly-supervised neural document classification model for article categorization. We examine classification accuracy on different levels of categories, and find that the model performs well on the top-level category, but when we descend the categories, the performance declines. Besides, we also perform experiments on finding the limitation of document length for the model's component RNN model. As for the limitation of document length for RNN model, we propose a method that first extracts the key sentences of documents then applies to train the RNN model.

Keyword Weakly-supervised learning, Text classification, Wikipedia category, Neural classification, Key sentence extraction

1. Introduction

Wikipedia is a free multi-lingual online encyclopedia which is constructed by volunteers, having over 5.7 million articles in English.

Wikipedia Category Structure In Wikipedia, articles are organized into hierarchical categories. Generally, a category in Wikipedia has three kinds of forms: (1) category consists of member articles, (2) category consists of subcategories, and (3) category consists of member articles and sub categories.

Figure 1 A category consists of subcategories and member articles

Wikipedia Article Categorization the task of Wikipedia categorization is that given a Wikipedia article, finding which category it should belong to. Our task is a kind of document categorization and our method can be applied to any categorization. In this paper, we consider the depth from the top category as a parameter of hardness of categorization.

Neural Networks In recent years, neural networks have been widely applied to NLP tasks such as machine translation, relation extraction, word embedding, etc, showing strong expressive power. But neural networks need a large volume of labeled training corpus to train a neural network, but member articles of one Wikipedia category are not sufficient to train a neural network. Yu et al. [6] proposed a weakly supervised neural model, called WSTC, which utilizes CNN [3] (Convolutional Neural Networks) and HAN [8] (Hierarchical Attention Networks) can work well on text classification, even if only a subset have class labels. HAN is a type of RNN, and it uses sequences of words and sentences, but the length of documents may exceed the learning ability of RNN. Therefore, in this paper we explore how the document length influences the performance of WSTC. Considering that a large amount of Wikipedia articles are long but one of the WSTC component models— the HAN model has limitation on handling long articles, we propose a method that we first utilize TextRank [5] to extract key sentences for each document, then apply them to the model.

Top-level categories, topic of each category is quite diverse in topics, such as sports, arts, and law, which are relatively simple to distinguish. But as we descend the category level, the topics of categories become detailed and similar to each other. Thus we also examine the classification accuracy on different levels of categories.

Our work makes four main contributions: First, we utilize apply the weakly-supervised neural model WSTC [6] to Wikipedia article categorization problem; second, we study on how document length influence the performance of WSTC; and third, we examine the classification accuracy on different level of categories; and fourth, we propose a method that combines key sentence extraction and the WSTC model and we find this model outperformed all other models.

The rest of this paper is organized as follows. Section 2 covers related work. Section 3 describes Wikipedia category structures and how we collect the dataset. Section 4 discusses combination of WSTC and text extraction. Section 5 present result and finding of result. Section 6 concludes this paper.

2. Related Work

There exist several researches related to Wikipedia article categorization. Aniket et al. [2] provide the first comprehensive quantitative mapping of the distribution of topics in Wikipedia based on Wikipedia category. Mohamed et al. [1] explore "is a" taxonomy from Wikipedia category graph. There has been a large amount of research on neural text classification and weakly supervised classification. Siwei et al. [4] proposed Recurrent Convolutional Neural Networks model which combine CNN and RNN model for text classification. Yangqiu et al. [7] take advantage of category name and project the name and document into the same sematic space. Then they perform classification based on the semantic similarity between the category surface name and the documents.

3. Wikipedia Category Structure and Data Collection

In this section, we discuss the Wikipedia category structure and how we collect our data for experiments.

3.1 Wikipedia category structure

In Section 1, we have introduced the three forms of categories — consisting of member articles, consisting of subcategories and consisting of member articles and subcategories. In this section, we describe the features of the top-level structure of Wikipedia categories. The top-level category is a category named *Contents*, which has subcategories of various types of encyclopedia contents, such as articles and glossaries, contents that assist navigation in the encyclopedia, such as indexes and portals, as well as pages related to the maintenance of the encyclopedia. However, since we focus only on the articles of Wikipedia, we select *Articles*, which is the highest level category for all articles of Wikipedia, as our top category. *Articles* have a subcategory named *Main topic classification* which can reach any article and categories and reflecting different fields of knowledge. Therefore, category *Main topic classification* is a list of Wikipedia's major topic classifications and contains 28 topics in total —*Arts*, *Concepts*, *Business*, *Culture* and so on.

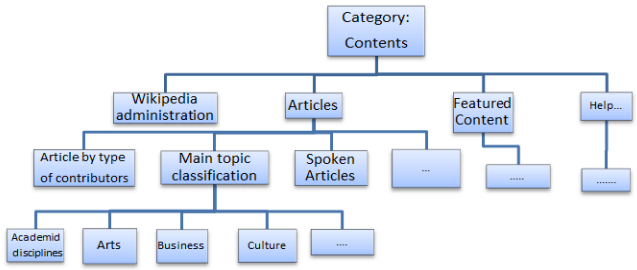


Figure 2 Overview of category system
When a node is a child of other node means this category is subcategory of the other category

3.2 Data Collection

In Section 3.1 we have discussed that Wikipedia articles are divided into the 28 top topics. But, those topics are not independent of each other. Also, certain topics are subcategories of multiple topics such as *Arts*, *Language*, *Society*, and *Culture*. Furthermore, certain topics share the same part of subcategories of *Culture*, and *Humanities* shares subcategories with *Arts*, and *Books*. In our case, we want to extract categories that have no inclusion relationship, no overlapping subcategories and no overlapping member articles. As a result, we extract the following five categories as top level categories – *Arts*, *Mathematics*, *Law*, *Language* and *Sports*. Because Wikipedia needs to control the size of member articles of one category, articles in a large category are often moved to subcategories. So, we extract its subcategory's member articles as this category's member articles. Finally, we obtained 7261 articles.

To examine whether or not the depth of the categories can cause degradation of classification accuracy, we extract articles from different levels of categories.

Table1 Number of articles for each category

Dataset name	Category(No. of articles)	Total
L1 Dataset	*Arts(843), Mathematics(2236),Law(1437),Language(1310),Sports(1427)*	7261
L2 Dataset	*Architecture (2390), Communication design(888), Culinary Arts(268), Fashion design(1089), Performing arts(1202),Perfumery(540)*	6377
L3 Dataset	*Advertising*(1024), *Document(1452), Graphic design(1258), Heraldry(766), Information visualization(201), Multimedia(642), Political communication(453)*	5823

Category *Arts*, which is a member of the top level categories, can divide into seven sub-topics: *Architecture, Communication design, Culinary Arts, Fashion design, Performing arts, Perfumery*. We regard those six topics as the second level categories. Then the total number of articles is 6377.

Similarly, *Communication design* can be divided into seven sub-topics – *Advertising, Documents, Graphic design, Heraldry, Information visualization, Multimedia, Political communication*. We regard those seven topics as the third level. The total number of the articles is 5823.

Notice that, for each category, we extract this category's member articles, its subcategories' member articles, subcategories of its subcategories member articles as its final member articles.

4. Weakly Supervised Text Classification Neural Model

Yu et at. [6] propose a weakly supervised text classification model which can apply deep learning to text classification problems, even there are only limited seed information such as class labels, and/or labeled articles. In this paper, we utilize this model to our Wikipedia article categorization problem. In this section, we first give a formulation of our task and then introduce the weakly supervised text classification (WSTC) neural model, then we discuss about the neural models – CNN and HAN that we use; finally we introduce how we combine TextRank and the WSTC-HAN model to deal with classification of long documents.

Table 2 Notation table

Notation	Description
$D=\{D_1,D_2,\ldots, D_n\}$	A set of Wikipedia articles
$C=\{C_1,C_2,..C_m\}$	m target categories
$L=\{L_1,L_2,..L_m\}$	Surface name for each category
$D_j^L=\{D_{j,1},\ldots D_{j,l}\}$	A set of l labeled documents in category.

4.1 Wikipedia Article Categorization Problem

The Wikipedia article categorization problem is that given a set of Wikipedia articles D, target categories C, we assign to each Wikipedia article D_j a category C_i.

4.2 Weakly Supervised Text Classification Model

Here, we discuss how we use the WSTC model to our task. The WSTC model can be divided into seven steps:

Extracting class related keywords. First we train word embeddings for each corpus. After we obtain word embeddings, we extract class related keywords. If the seed information is label surface name L, then for each class j, the embedding of surface name L_j of each category C_j is used to retrieve top-t nearest words in the semantic space, and these t words are adopted as class related keywords.

If the seed information is a small amount of Wikipedia articles whose category labels are known, we first extract t representative keywords from the labeled articles by using tf-idf weighting, then adopt these words as class-related keywords.

Modeling the semantic distribution of each class as von Mise Fisher (vMF) After extracting class related keywords, we model the word distribution of each class by von Mise Fisher (vMF), such that the probability distribution of each class is given by:

$$f(x;\mu,k)=c_p(k)e^{k\mu^T x} \quad (1)$$

Here, $k>0$, $||\mu||=1$, $p\geq 2$, and $c_p(k)$ is normalization given by:

$$c_p(k)=\frac{k^{\frac{p}{2}-1}}{2\pi^{\frac{p}{2}}I_{\frac{p}{2}-1}(k)} \quad (2)$$

Where $I_r(.)$ is the modified Bessel function of the first kind at order r. Let X be a set of vectors for keywords on unit sphere,

$$X=\{x_i|\ x_i \text{ drawn from } f(x;\mu,k), 1\leq i\leq t\}$$

vMF needs two parameters k and μ ,where μ is calculated by the following maximum likelihood estimattion:

$$\mu=\frac{\sum_{i=1}^{t}x_i}{||\sum_{i=1}^{t}x_i||} \quad (3)$$

and

$$\frac{I_{p/2}(k)}{I_{\frac{p}{2}-1}(k)} = \frac{||\sum_{i=1}^{t} x_i||}{t} \qquad (4)$$

Generating pseudo documents. In order to pre-train the neural model, we first generate pseudo documents from the vMF distribution. To generate pseudo documents, a document vector is chosen from the class probability $(x; \mu, k)$, then pseudo document d_i is generated by drawing words w from the probability distribution:

$$P(w|d_i) = \begin{cases} \alpha P_B(\mathrm{w}) & \mathrm{w} \notin V_{d_i} \\ \alpha P_B(\mathrm{w}) + (1-\alpha)\frac{\exp(d_i x_w)}{\sum_{w' \in V_{di}} \exp(d_i x_{w'})} & \mathrm{w} \in V_{di} \end{cases} \quad (5)$$

Here, $P_B(w)$ is the background distribution of word w, α is background distribution weight, and V_{di} is the keywords consisting of the most similar ξ words in the semantic space for document d_i. For each category β documents are generated.

Neural model pre-training In order to overcome the overfit problem, this model creates pseudo labels for pseudo documents rather than using one-hot encoding method. The pseudo label ℓ_i for pseudo documents D_i^* is given as:

$$\ell_{ij} = \begin{cases} (1-\alpha) + \frac{\alpha}{m} & D_i^* \text{ is generate from class j} \\ \frac{\alpha}{m} & \text{otherwise} \end{cases} \qquad (6)$$

Then the neural model is pre-trained by minimizing the KL loss between output Y and pseudo labels L. The KL loss is given as:

$$\mathrm{KL}(L||Y) = \sum_i \sum_j l_{ij} log \frac{l_{ij}}{y_{ij}} \qquad (7)$$

Neural model self-training After pre-train the model, we get the Pre-WSTC model, we use unlabeled articles to self-train the Pre-WSTC model. It first uses the Pre-WSTC to categorize the unlabeled articles, and current output Y is obtained. Then the pseudo labels are updated by equation (8):

$$l_{ij} = \frac{y_{ij}^2}{\sum_{j'}(y_{ij}/f_j')} \quad , \qquad (8)$$

where $f_i = \sum_i y_{ij}$ is the soft frequency for class j. Pseudo labels are updated iteratively while minimizing the KL loss. When the number of changes in the category assignment is less than σ, this iterative process will stop and we can get our WSTC model.

4.3 Neural Models

In the neural model part, we use two kinds of neural models—CNN and HAN.

4.3.1 CNN Model. Yoon al. [6] proposed applying the CNN model to the sentence classification problem. In our case, the input of CNN is a Wikipedia article of length l_e which is represented by a concatenation of word vectors:

$$\mathrm{d} = x_1 \oplus x_2 \oplus \ldots . \oplus x_{le}, \qquad (9)$$

where $x_i \in R^p$ is the p-dimensional word vector of the i^{th} word. A feature f_i is generated with window size s:

$$f_i = f(w \cdot x_{i:i+s-1} + b) \qquad (10)$$

Where $w \in R^{hp}$ is filter operating, $b \in R$ is bias terms. After the filter operation, a feature map is generated as

$$f = [f_1, f_2, \ldots f_{le-s+1}] \qquad (11)$$

Finally, max-pooling operation is performed for each feature map to obtain final feature and the fully connected layer of those final features is the input of a softmax layer, from which the final output is generated.

4.3.2 HAN Model. Zichao et al.[8] proposed the HAN model which takes advantage of the hierarchical architecture of documents by using a sequence encoder and attention layer. The input article is represented by a sequence of sentences s_i, $i \in [1, L]$ and each sentence is represented by a sequence of words w_{it}, $t \in [1.T]$. For the t -th word, the GRU calculates the new states by:

$$h_t = (1 - z_t) \odot h_{t-1} + (z_t) \odot h_t^* \quad (12)$$

r_t is the reset gate vector :

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \qquad (13)$$

z_t is the update gate vector :

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \qquad (14)$$

h_t^* is the candidate state vector :

$$h_t^* = thah(W_h x_t + r_t \odot U_r h_{t-1} + b_h), \quad (15)$$

where x_t is the t-th sequence vector. After encoding words, we apply the attention layer to generate the sentence level representation. Similarly, after the sentence level, the attention layer generates the document level representation, which becomes the input for the softmax layer. The final output is generated from the softmax layer.

4.4 Combination of WSTC-HAN and TextRank

RNN-based models such as HAN have difficulties in dealing with long sequences, so one method to resolve this problem is to compress long articles into short version. The simplest way to compress an article is to use the leading sentences of one article. But important contents may appear in any position of the article, so only using the leading sentences may not catch certain import information. So, in this paper, we consider another way to compress one article. We use TextRank to compress one article by extracting key sentences.

TextRank [5] is first proposed by Mihalcea, which can be

Table 3 Dataset Statistics

name	Category name(no. of articles)	train dataset	test dataset	Average document length(words)	Average document length(sentences)
L1 dataset	*Arts*(843), *Mathematics* (2236),*Law*(1437), *Language*(1310), *Sport*(1427)	5808	1453	1325	52
L2 dataset	*Architecture*(2390), *Communication design*(888), *Culinary arts*(268), *Fashion design*(1089), *Performing arts*(1202), *Perfumery*(540),	5104	1273	976	42
L3 dataset	*Advertising(1024), Documents(1452), Graphic design(1258), Heraldry(766), Information visualization(201), Multimedia(642), politic communication(453)*	4661	1162	969	41

used to extract keywords and key sentences from a single document without supervised learning. TextRank also has been applied to a wide range of areas such as text summarization. In this paper, we utilize TextRank to extract key sentences. TextRank is a graph-based algorithm, where each sentence is represented as a graph node, and an edge between two nodes is weighted by similarity:

$$Similarity(s_i, s_j) = \frac{|\{w_k | w_k \in s_i \& w_k \in s_j\}|}{log^{|s_i|} + log^{|s_j|}} \quad (16)$$

$$w_{ij} = Similarity(s_i, s_j) \quad (17)$$

And score of each sentence is calculated by:

$$Score(V_i) = (1-d) + d * \sum_{v_j \in In(V_i)} \frac{w_{ij}}{\sum_{v_{k \in out(v_j)}} w_{jk}} Score(v_j) \quad (18)$$

Where $In(v_i)$ is the set of vertexes that point to v_i(predecessors), $Out(v_i)$ is the set of vertexes that v_i points to(successors). Then calculate each vertex's score until convergence.

After we calculate the score for each sentence, we rank sentences of one article according to the score and finally we choose top-t sentences as the compressed version of one article. As the result, we use compress version of documents for pre-training and self-training WSTC-HAN model.

5. Experiments

5.1 Evaluation method

In this paper, we use F1-Micro and F1-Macro score to evaluate the performance of the model. Suppose TP represent the true positive, FP means false positive, FN means false negative, TN means true negative.

F1-micro first calculate all classes' TP, FP, FN and TN, then calculate F1 score whereas F1-macro first calculate the F1 score of each class then regard the average of F1 score as F1-macro score.

5.2 Parameter setting

First, the number of labeled articles for each class is 20, if the seed information is labeled article, the number t of extracted keywords for each category is 10. The background word distribution weight α is 0.2. The number of pseudo articles for each category β is 500. The self-training stop condition is $\sigma = 0.0001$. The dimension of word embeddings is 100.

5.3 Dataset Statistics

In Section 3 we have discussed about how we collect the datasets – top-level dataset, second level dataset, and the third level dataset. For each dataset, we split it into training dataset and test dataset, according to the radio of 8:2. Besides, as the category level goes down, the topics between them become more similar, so there exist several Wikipedia articles that belong to multiple categories. But we only need Wikipedia articles that only belong to one category, so we remove articles which belong to two or more categories. In Table 4 we will show the number of overlapped articles in each dataset.

Table 4 Overlap articles statistic (L2 dataset)

Name of each category	Overlap articles for each class
Architecture	0
Communication design	2
Culinary arts	0
Fashion design	4
Performing arts	2
Perfumery	2
total	8

Table 5 Overlap articles statistic (L3 dataset)

Name of each category	Overlap articles for each class
Advertising	0
Documents	1
Graphic design	10
Heraldry	7
Information visualization	2
Multimedia	9
Political communication	19
Total	48

Besides, in this paper, we utilize two kinds of seed information—category surface names and labeled Wikipedia articles. If the seed information is category name, we directly regard the category surface name as class related keywords, if the seed information is labeled articles, we extract t=10 keywords for each class and regard them as class related keywords.

Table 6 class-related keyword of different seed information for L1 dataset

Category surface name	Labeled Wikipedia article
arts	art, video, design, work, artist…
mathematics	filter, permutation, crank, additive…
law	grant, monarch, felony, equity, legislature
language	learner, jargon, pronunciations, progmatics…
sports	footbag, polo, basketball, rink…

Table 7 class-related keyword of different seed information for L2 dataset

Category surface name	Labeled Wikipedia article
architecture	buildings, architectural, Audubon, townhouses, …
communication, design	webpage, messaging, telephony smartphone …
culinary, arts	foods, chicken, fried, seafood dishes…
fashion, design	sarafpour, wendi, turunen, womenswear …
performing, arts	demonstrations, theatricality, ipcny, musicology …
perfumery	fragrances, gourmand, scents, perfume …

Table 8 class-related keyword of different seed information for L3 dataset

Category surface name	Labeled Wikipedia article
advertising	advertising, ad, campaign, television…
documents	letters, cards, recipient, …
graphic, design	graphics, visual, typography, lines,…
heraldry	arms, coat, heraldry, royal …
information , visualization	computer, graphics, research, visualization …
Political, communication	Political, committees, agreement
multimedia	tv, digital, channels, video …

5.4 Experimental Results

In this section, we report our results and what we find in the results.

5.4.1 Effect of category level. In this paper, we apply the WSTC model on datasets L1, L2 and L3 of three different category levels. The results are shown in Figures 3 and 4. Dataset L1, extracted from the top-level category, shows better results than L2 and L3 from lower categories, for both CNN and HAN models, and in both macro- and micro-averages. One explanation is that when the category level descends, the topics between each category become more similar. For example, for extracted keywords for each category of dataset L1, there is no overlapping word, but for dataset L3, the keywords for category graphic design and information visualization have a common keyword – "graphics", indicating that the topics between them become more similar, making the classification task harder.

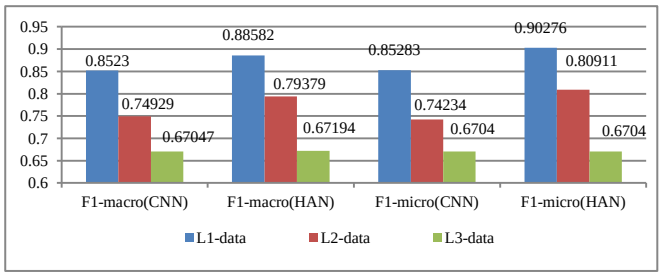


Figure 3 Classification results on different levels of category hierarchy. The seed information is labeled articles.

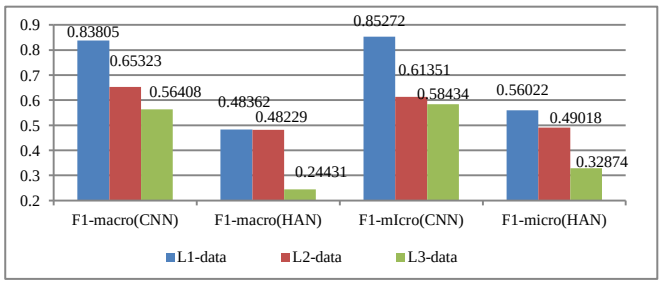
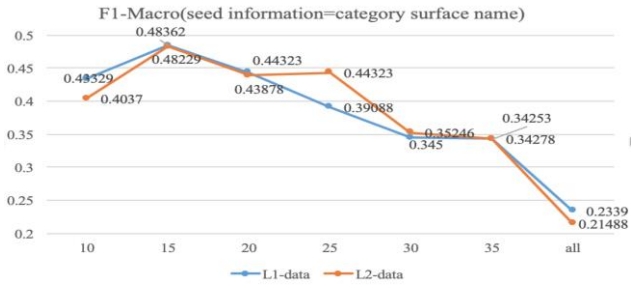


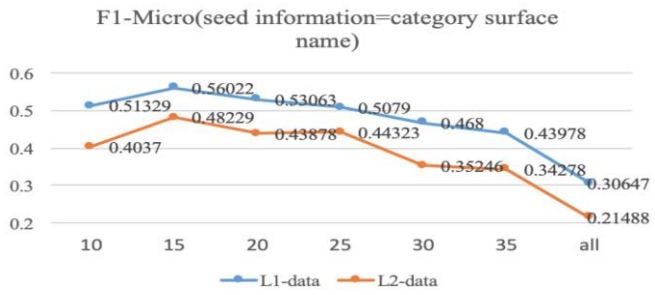
Figure 4 Classification results on different levels category hierarchy. The seed information is category surface name.

5.4.2 Effect of article length. In this paper, we apply HAN

which is a variation of RNN models that takes advantage of text sequences. The HAN model updates a state according to the state of one previous state, so if the length of article is too long, the information which is carried by the beginning of the article will contribute less to the model. Besides, as the length of articles increase, the training time will also become longer. So, it is necessary to find a proper length of articles which can maximize the F1 score at the same time decreasing training time. So, we explore how varying lengths of articles affect the performance of the model. We explore seven different lengths – the leading 10, 15, 20, 25, 30, 35 sentences, and the whole documents. In the experiments we find that when the document length is trimmed to around leading 15 sentences, the best F1 score is achieved while the training time is much smaller than the time used for the whole documents. Also, Figure 5 shows when the seed information is category name, we can see the best F1-micro and -macro scores at length 15, for both L1 and L2. Figure 6 shows when the seed information is labeled articles, the best F1-micro and F1-macro scores at length 15, for both L1 and L2.

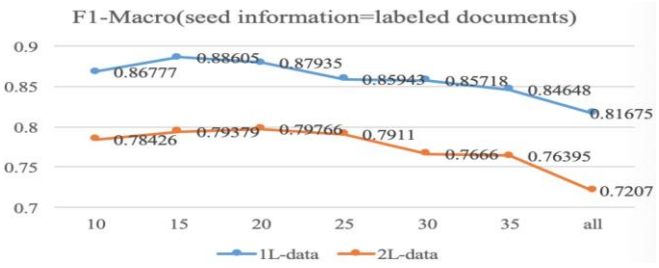


(a)

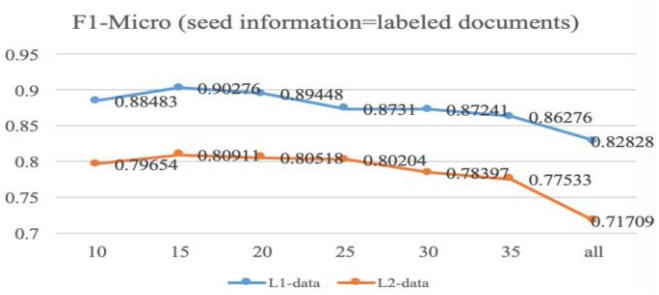


(b)

Figure 5 (a) F1-macro score on different lengths of documents (b) F1-micro score on different lengths of documents, where the seed information is category name.



(a)



(b)

Figure (a) F1-macro score on different lengths of documents; (b) F1-macro score on different lengths of documents, where the seed information is labeled articles.

Figure 7 shows the training time at varying lengths of articles, where the seed information is labeled documents and the dataset is top-level L1. As the length of articles increases, the training time is increasing. When the article length is 15 sentences, training time is 3260.77 sec. On the other hand, if we use the whole article, the training time is 4735.79 sec, which takes 1474.33 sec more than when document length is 15 sentences.

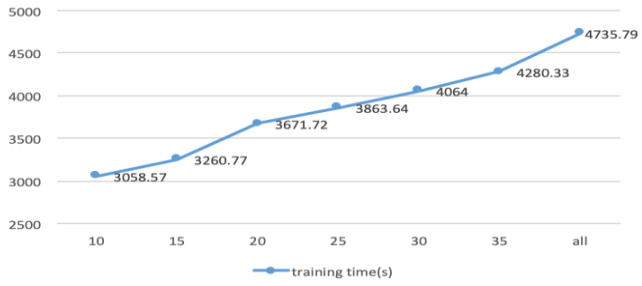


Figure 7 Training time for different length of documents for L1 dataset. X-axis is document length, and Y-axis is training time.

5.4.3 Influence of word dimension for WSTC-HAN. In the previous experiments, we perform the experiments where word dimension is 100. In this experiment, we show the results on varying word dimension and varying document length, for the WSTC-HAN model. The document length is varied within the leading 5, 10, 15, 20, 25, 30

Table 9 Influence of different word dimensions of different document length for WSTC-HAN model

Word dimension \ Document length	F1-score	5	10	15	20	25	30	all
50	F1-Macro	0.8515	0.86113	**0.86883**	0.85432	0.85513	0.85258	0.82029
	F1-Micro	0.86376	0.87696	**0.88367**	0.87956	0.87238	0.87235	0.83108
100	F1-Macro	0.873	0.86777	**0.88605**	0.87935	0.85943	0.85718	0.81675
	F1-Micro	0.889	0.88483	**0.90276**	0.89448	0.8731	0.87241	0.82828
200	F1-Macro	0.85827	**0.86145**	0.85421	0.85707	0.85397	0.8536	0.83044
	F1-Micro	0.87655	**0.87931**	0.87586	0.87172	0.87103	0.87103	0.83259
300	F1-Macro	0.86635	**0.86935**	0.86662	0.86543	0.85612	0.85258	0.82931
	F1-Micro	0.88332	**0.88728**	0.88298	0.88278	0.87263	0.87235	0.83592

Table 10 F1 score for all methods on three datasets. The seed information is labeled documents.

Model	L1 dataset		L2 dataset		L3 dataset	
	F1-Macro	F1-Micro	F1-Macro	F1-Micro	F1-Macro	F1-Micro
WSTC-CNN	0.8523	0.85283	0.74929	0.74234	0.67047	0.6704
WSTC-HAN	0.81675	0.82828	0.7207	0.71799	0.62344	0.62198
WSTC-HAN(leading 15 sentences)	0.88582	0.90276	0.79397	0.80911	0.67194	0.6704
WSTC-HAN(TextRank)	**0.894**	**0.90966**	**0.80686**	**0.81697**	**0.68221**	**0.68675**

sentences and all the sentences of documents. The seed information is labeled documents. We find that the best F1-score is achieved when the word dimension is 50 and 100, and the document length is first 15 sentences. When the word dimension is 200 and 300, the best result is achieved with the document length 10 sentences. Besides, word dimension 100 is showing the overall best result, surpassing dimensions 50, 200, and 300.

5.4.4 Results on combination of WSTC-HAN and TextRank. In this experiment, we compare the results of different models where the seed information is labeled documents. Table 10 shows that when TextRank is used to extract top 15 key sentences, WSTC-HAN outperforms all the other models. If we only adopt the leading 15 sentences as the compressed version of one article, WSTC-HAN cannot catch important information in the middle or end of articles. On the other hand, TextRank identifies connections between various sentences, and promotes sentences that contain words frequently mentioned throughout the article. So, instead of only using the leading part of articles, we use TextRank to capture sentences from the whole article, that are effective for generating pseudo documents for self-training of neural document categorization.

6. Conclusion and Future Work

In this paper, we discussed weakly supervised neural models for categorizing long articles of Wikipedia. We revealed that as the category level descends, the classification problem becomes harder and the accuracy declines. We also explored how document lengths affect the results, and find that when a proper length for document curtailing is used, it can both increase the accuracy and decrease the training time. Finally, we proposed a method that combines summarization by TextRank and WSTC-HAN, outperforming all other models. Generally, document categorization becomes harder as category topics become similar to each other. So for future work, we try to deal with overlapping phrases.

References

[1] M. B. Aouicha, M. Ali, H. Taieb and M. Ezzeddine, “Derivation of ‘is a ’ taxonomy from Wikipedia Category Graph” Eng.Appl.Artif.Intell. pp.265-286 2016

[2] A. Kittur, “What’s in Wikipedia? Mapping Topics and Conflict Using Socially Annotated Category Structure”, ACM. pp.1509-1512, 2009

[3] Y. Kim, “Convolutional Neural Network for Sentence Classification”, EMNLP. pp. 1746-1751, 2014

[4] S. Lai, L. Xu, K. Liu, and J. Zhao, “Recurrent Convolutional Neural Networks for Text Classification”. AAAI. Vol.333, 2015

[5] R. Mihalcea and P. Tarau, “TextRank: Bridging Order into Texts”. EMNLP. pp.404-411, 2004

[6] Y. Meng , J. Shen, C. Zhang and J. Han, “Weakly-Supervised Neural Text Classification”, CIKM 2018.

[7] Y. Song and D. Roth, “On Dataless Hierarchical Text Classification”. AAAI, 2014

[8] Z. Yang, D. Yang, C. Dyer, X. He, A. J. Smola, and E. H. Hovy. “Hierarchical Atthention Neyworks for Document Classification” HLT-NAACL 2016