| 저자 (Authors) | Hwiyeol Jo, Jin-Hwa Kim, Kyung-Min Kim, Jeong-Ho Chang, Jae-Hong Eom, Byoung-Tak Zhang |
| 출처 (Source) | 정보과학회 컴퓨팅의 실제 논문지 23(5), 2017.5, 322-327 (6 pages) |
| 발행처 (Publisher) | KOREA INFORMATION SCIENCE SOCIETY |
| URL | http://www.dbpia.co.kr/Article/NODE07164901 |
| 이용정보 (Accessed) | 서울대학교 147.46.78.*** 2017/05/29 13:48 (KST) |
깊은 신경망 기반 대용량 텍스트 데이터 분류 기술
(Large-Scale Text Classification with Deep Neural Networks)

조휘열†  김 진 화**  김경민***
(Hwiyeol Jo)  (Jin-Hwa Kim)  (Kyung-Min Kim)
장정호****  엄재홍*****  장병탁******
(Jeong-Ho Chang)  (Jae-Hong Eom)  (Byoung-Tak Zhang)

요약
문서 분류 문제는 오랜 기간 동안 자연어 처리 분야에서 연구되어 왔다. 우리는 기존 컨볼루션 신경망을 이용했던 연구에서 나아가, 순환 신경망에 기반을 둔 문서 분류를 수행하였고 그 결과를 종합하여 제시하려 한다. 컨볼루션 신경망은 단층 컨볼루션 신경망을 사용했으며, 순환 신경망은 가장 성능이 좋다고 알려져 있는 장기 단기 기억 신경망과 회로형 순환 유닛을 활용하였다. 실험 결과, 분류 정확도는 Multinomial Naïve Bayesian Classifier < SVM < LSTM < CNN < GRU 순서로 나타났다. 따라서 텍스트 문서 분류 문제는 푸즈의 고려하는 것보다는 문서의 feature 추출에 분류하는 문제에 가깝다는 것을 확인할 수 있었다. 그리고 GRU가 LSTM보다 문서의 feature 추출에 더 적합하다는 것을 알 수 있으며, 적절한 feature와 푸즈 정보를 함께 활용할 때 가장 성능이 잘 나온다는 것을 확인할 수 있었다.

키워드: 딥러닝, 대용량 문서 분류, 자연어 처리, 인공신경망

Abstract
The classification problem in the field of Natural Language Processing has been studied for a long time. Continuing forward with our previous research, which classifies large-scale text using Convolutional Neural Networks (CNN), we implemented Recurrent Neural Networks (RNN), Long-Short Term Memory (LSTM) and Gated Recurrent Units (GRU). The experiment’s result revealed that the performance of classification algorithms was Multinomial Naïve Bayesian Classifier < Support Vector Machine (SVM) < LSTM < CNN < GRU, in order. The result can be interpreted as follows: First, the result of CNN was better than LSTM. Therefore, the text classification problem might be related more to feature extraction problem than to natural language understanding problems. Second,
judging from the results the GRU showed better performance in feature extraction than LSTM. Finally, the result that the GRU was better than CNN implies that text classification algorithms should consider feature extraction and sequential information. We presented the results of fine-tuning in deep neural networks to provide some intuition regard natural language processing to future researchers.

**Keywords:** deep learning, large-scale text classification, natural language processing, artificial neural networks

1. **Introduction**

1.1 **Classification Problem**

Lots of data is generated and transmitted through media (e.g. internet news, web page), messengers, and so on. The size of data increases rapidly as well as simultaneously, so the process of determining which category the data belongs to, and deciding whether it is useful or not has become more important.

In the same context, the classification problem is related to data mining, which is to find meaningful information, (such as relationships between data) in large-scale data [1]. Video, music, and data of other formats can be easily represented in text format, so this classification method will be helpful in sorting out meaningful information from the mass of data.

1.2 **General approaches on Classification Problem**

A traditional text classification process can be summarized as in Figure 1.

The inputs are raw text data written in natural language (e.g. English, Korean, Japanese, and Chinese). First, we have to do pre-processing to modify our data to fit our chosen classifier. The most frequently selected classifiers are Multinomial Naïve Bayes Classifier [2,3], Support Vector Machine [4], and Neural Networks including deep architectures such as Convolutional Neural Networks [5] and Recurrent Neural Networks [6].

In our experiment, we collected 623,303 news articles from web with enough words to represent a topic. We divided this data into 9 large categories and 68 small categories. Our data is composed of Korean documents parsed by morpheme while other research has been focused on English data.

2. **Background**

2.1 **Multinomial Naïve Bayes**

Naïve Bayes classifier [2] is a simple but strong probabilistic classifier based on Bayes’ theorem which assumes independence. Although an individual event may affect other events, simply assuming that the two events are independent, it expresses the joint probability of two events as just a product of two probabilities. While Naïve Bayes classifier refers to the conditional independence of the features in the model, Multinomial Naïve Bayes classifier uses a multinomial distribution for the features.

2.2 **Support Vector Machine**

SVM [4] is a strong and widely used classifier which gets its classification boundary by changing the boundary to the greater distance (margin) between each of the class. That is, SVM maximizes the distance between the boundary and the data vector, which improves the accuracy when new data comes, minimizing its generalization error. The shortest distance of each class to data is called margin, and the data vector with the minimum distance from the boundary is called the support vector.

2.3 **Convolutional Neural Networks (CNN)**

CNNs [5] are a hierarchical model, inspired by the human’s neural network structure, used in various pattern recognition problems. It operates two layers, convolutional layers and subsampling layers (also called max-pooling layers), sequentially. Finally, before the output layer, it uses fully connected layers. Convolutional layer employs filter banks to input images, performing 2D-filtering. Subsampling layers extract a local maximum value from the convolutional layers, mapping to 2D-images. While widening the region
gradually, the layer repeats down-sampling. Lastly, using fully-connected layer, the model learns the way to minimize input-out errors through back-propagation. CNNs have been widely used to solve problems in the field of vision (e.g. face recognition, handwriting recognition), but recently it is tried on Natural Language Processing (NLP).

CNNs on NLP, the first layer makes word vectors by referencing a lookup table. Considering each word as a pixel in images, CNN represents each document by \(|\text{document}| \times 1\) vectors where the size of a channel is equal to the number of extracted words per document. Then, the model progresses similarly to the case of images, using the word vectors as feature vectors.

2.4 Recurrent Neural Networks (RNN)

RNNs [6] have been used for sequence learning. Despite initial successful results, it was difficult to learn a model with RNN, so the work to improve the basic architecture continues. The most successful example among RNNs is Long–Short Term Memory (LSTM) [7], which can store and return long sequence information through explicit gating mechanisms with built-in error carousels. LSTM is a model designed to address the issue of vanishing gradient in RNN memory gating mechanism. With LSTM unit, the gradient can be efficiently propagated, and the model can learn the long-term dependency. Recently, a new variation on RNNs is on the rise such as Gated Recurrent Units (GRUs) [8], which is a simplified version that reduces the number of parameters in LSTM, using the similar gating mechanism. GRU consists of reset gates that determine which part of the old memory will be preserved and update gates that set a new value to memory cells. GRU can be interpreted as a model that combines the content-based soft attention mechanism with LSTM.

3. Dataset

For the experiment, we collected 623,303 news articles from the internet in various fields, allocating 70%, 15%, 15% of the data to be used for training, validation, and test data respectively. The distribution of data is presented in Table 1.

The dataset has two-level hierarchy. We defined super category as large–category and subcategory as small–category. The average number of documents is 9,166, and the deviance is 15,460. It is because we crawled data during a specific period without considering the number of documents in each topic.

4. Experiment

4.1 Design

We implemented the model using Python 2.7 for preprocessing and torch7 with nn and rnn packages [9]. To implement CNN, we first parsed text documents by morpheme, building a lookup table for top-n frequency words. After that, we slid kernels, which

<table>
<thead>
<tr>
<th>Large Category</th>
<th>Small Category(#Document)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Science</td>
<td>Kistiscience(2,065), Science_general(3,063), Scienceskill(425)</td>
</tr>
<tr>
<td>Special Section</td>
<td>Esc_section(8,494)</td>
</tr>
<tr>
<td>International</td>
<td>Arabafrica(6,221), China(5,556), Europe(5,431), Global_economy(2,199), International_unit(564), Asiapacific(3,984), America(14,359), Japan(6,437), International_general(14,203), Global_topic(887)</td>
</tr>
<tr>
<td>Economy</td>
<td>Working(2,852), Finance(6,228), Marketing(1,098), Heri_review(795), IT(3,998), Car(3,371), Stock(4,299), Biznews(6,393), Consumer(5,160), Property(5,970), Economy_general(49,679)</td>
</tr>
<tr>
<td>Politics</td>
<td>Bluehouse(5,856), Assembly(11,946), Defense(16,864), Politics_general(35,056), Administration(1,951), Diplomacy(2,936)</td>
</tr>
<tr>
<td>Culture</td>
<td>Travel(889), Movie(7,761), Book(14,098), Culture_general(12,067), Entertainment(13,054), Music(7,761), Religion(2,218), Skypestar(50), Novel_salt(126), Baebyprincess(126)</td>
</tr>
<tr>
<td>Society</td>
<td>Labor(6,305), Internalmove(459), Women(1,604), Religious(2,448), Ngo(2,332), Media(6,128), Campus(4,338), Environment(5,249), Society_general(105,143), Schooling(19,725), Health(7,484), Handicapped(797), Rights(2,232), Obityuray(4,398), Area(52,450)</td>
</tr>
<tr>
<td>Opinion</td>
<td>Dica(1,311), Column(18,701), Because(3,462), Editorial(9,398), Argument(141)</td>
</tr>
<tr>
<td>Sports</td>
<td>Sports_general(24,113), Gameschedule(2,672), Baseball(11,911), Baduk(605), Scoreboard(382), Soccer(17,042), Golf(3,834)</td>
</tr>
</tbody>
</table>

[Total - 9 / 68 classes] 623,303
learns appropriate parameters. The column size of the kernel was equal to word embedding size whereas the row was a hyper-parameter. We used ReLU as the activation function and then used logSoftMax function to output the probability that the document belongs to each 9 or 68 categories. We picked the highest value as the topic of the classified document [10].

We also used RNN as a classifier. GRU showed the best performance among them. Like CNN, we built a lookup table, split it by each word vector, and then used the vectors as inputs for GRU. By training GRU, the lookup table was updated and the last hidden vector was used as the output. Same as above, ReLU was used with logSoftMax function. Finally, the highest probability vector was chosen as a topic [11].

The process of CNN and RNN can be summarized as in Figure 2 and Figure 3.

4.2 Results
The performance of classification algorithms is presented in Table 2 and Table 3.

Table 2 The accuracy of large-category classification

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy (Top-1,3,5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNB</td>
<td>0.641 0.911 0.958</td>
</tr>
<tr>
<td>SVM</td>
<td>0.795 0.960 0.991</td>
</tr>
<tr>
<td>CNN</td>
<td>0.856 0.986 0.997</td>
</tr>
<tr>
<td>LSTM</td>
<td>0.811 0.965 0.994</td>
</tr>
<tr>
<td>GRU</td>
<td>0.886 0.992 0.999</td>
</tr>
</tbody>
</table>

Table 3 The accuracy of small-category classification

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy (Top-1,3,5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNB</td>
<td>0.399 0.679 0.794</td>
</tr>
<tr>
<td>SVM</td>
<td>0.614 0.851 0.906</td>
</tr>
<tr>
<td>CNN</td>
<td>0.700 0.920 0.962</td>
</tr>
<tr>
<td>LSTM</td>
<td>0.670 0.895 0.942</td>
</tr>
<tr>
<td>GRU</td>
<td>0.725 0.937 0.971</td>
</tr>
</tbody>
</table>

4.3 Fine-tuning
To find which hyper-parameter is important, we fine-tuned our CNN and RNN models. We fixed word dictionary size to 100,000, then extracted 500 words per document (word_per_doc), fixed the word embedding size to 300, and decreased the learning rate from 0.2 to 0.001.

In CNN, we changed ChannelSize from 100 to 2,500, but we couldn’t observe a trend in accuracy. The slight difference in accuracy could come from random initialization at word embedding. The kernel-Size, which can be considered as an N-gram model, showed the best performance when 5. We experimented with activation functions such as ReLU, tanh, and sigmoid. In contrast to the well-known knowledge that ReLU shows the best performance, sigmoid function showed the best accuracy.

The test accuracy according to the size of hidden nodes is presented in Figure 4. We used early stop-
ping to avoid overfitting. The result showed that 500 hidden nodes are the optimal hyper-parameter in CNN and LSTM but not in GRU. More than 1,000 hidden nodes caused overfitting easily.

5. Conclusion

This research includes text classification using deep neural networks. We collected 623,303 news articles from the internet and classified them into 9 large-categories and 68 small-categories, respectively. The accuracy of algorithms can be listed as Multi-nominal Naïve Bayesian Classifier < SVM < LSTM < CNN < GRU in order. The result can be interpreted as follows: Firstly, the result of CNN was better than LSTM. Therefore, the text classification problem might be closer to feature extraction problem than to natural language understanding. Secondly, judging from the results the GRU showed better performance in feature extraction than LSTM. Finally, the result that GRU was better than CNN implies that text classification algorithms should consider feature extraction and sequential information. Furthermore, we presented the results of fine-tuning in deep neural networks to provide some intuition to future researchers in natural language processing.

References


장 정호
1995년 서울대학교 컴퓨터공학부 학사
2005년 서울대학교 컴퓨터공학부 박사

염재홍
1999년 강원대학교 컴퓨터공학과 학사
2001년 서울대학교 전기-컴퓨터공학 석사
2009년 서울대학교 컴퓨터공학부 박사

장병탁
정보과학회 컴퓨팅의 실제 논문지
제 23 권 제 2 호 참조