World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:14, No:11, 2020

Personal Information Classification Based on Deep Learning in Automatic Form Filling System

Shunzuo Wu, Xudong Luo, Yuanxiu Liao

Abstract—Recently, the rapid development of deep learning makes artificial intelligence (AI) penetrate into many fields, replacing manual work there. In particular, AI systems also become a research focus in the field of automatic office. To meet real needs in automatic officiating, in this paper we develop an automatic form filling system. Specifically, it uses two classical neural network models and several word embedding models to classify various relevant information elicited from the Internet. When training the neural network models, we use less noisy and balanced data for training. We conduct a series of experiments to test my systems and the results show that our system can achieve better classification results.

Keywords—Personal information, deep learning, auto fill, NLP, document analysis.

I. Introduction

NOWADAYS, the explosion of network data is dazzling with all kinds of information. The data on the Internet can be divided into structured data, semi-structured data, and unstructured data according to their forms. With the development of deep learning technology, machines can better find the regular, valuable and understandable knowledge contained in the data. Especially, by training neural networks, one can find the relationship between the data, then make predictions.

In daily office activities, filling various forms occupy a lot of office time, and there is a lot of repetitive work in a filling process, which greatly reduces the efficiencies in work. Although existing systems of automatic form filling can help office staff to fill some items automatically, they can do it only based on the existing knowledge. For example, DingTalk uses automatic systems of form filling. However, DingTalk can only obtain data from user input. And in the work of Friends of Scientific Research, they obtain structured data, and most of them are user input. To let a form filling system use the information from the Internet, in this paper, we integrate deep learning and natural language processing methods to extract effective information from unstructured data on the Internet.

The rest of this paper is organized as follows. Section II presents the architecture of our system. Section III uses an example to explain the operating process of our system. Section IV experimentally evaluates our system. Section V discusses related work. Finally, Section VI summarises this paper with future work.

Shunzuo Wu, Xudong Luo, and Yuanxiu Liao are with Guangxi Key Lab of Multi-Source Information Mining & Security, College of Computer Science and Information Engineering, Guangxi Normal University, Guilin 541004, China (e-mail: luoxd@mailbox.gxnu.edu.cn, liaoyuanxiu@mailbox.gxnu.edu.cn).

II. System Architecture

Fig. 1 shows the architecture of our system. First, we use a crawler to download the data from a target webpage. The crawled data are unstructured, asymmetric, and noisy. So, we preprocess the data by using methods such as removing stop words. We use the original corpus and word2vec to generate word vectors. Then we train a neural network to obtain a classification model. Next, we extract entities from the classified data to obtain more accurate and no redundant data. Finally, we use the natural language generation model to fill in a form for users.

A. Crawler

Fig. 2 shows the basic flow of the crawler. A crawler is a program or script that automatically grabs information from the World Wide Web in accordance with certain rules. Compared with the traditional manual collection of information, it is more efficient in terms of speed, saving, and reusability. There are the following types of crawlers: (1) general-type crawlers, also called a full-web crawler, which is mainly used in search engines, from the initial URL to the entire web page, but requires large storage capacity, fast speed requirements, and powerful work performance); (2) focused crawlers, which only focus on certain aspects of information (usually pre-defined), (4) incremental crawlers, which re-crawls to update at regular intervals, and (5) deep crawlers, which require verification to crawl.

Algorithm 1 is the crawler algorithm. The data collected in this paper is personal information data about a scholar, so we use a focused crawler. In the crawler algorithm, the input is the address of a website concerning a scholar, and the output is the personal information of the scholar. The process as follows. First, set an initial URL, then create the file for saving data. Second, send a request to the target page and get a response that includes the source code of the target page. Third, parse the source code and use a selector to select the document object that contains the document content to be obtained. Finally, save the information in the file and update the new URL until all pages are crawled.

The crawled data are in the three categories:

1) structured data,
2) semi-structured data, and
3) unstructured data.

Structured data exist on web pages in the form of key-value, and these data only need to be entity extracted. Semi-structured data and unstructured data exist in many forms, and only the

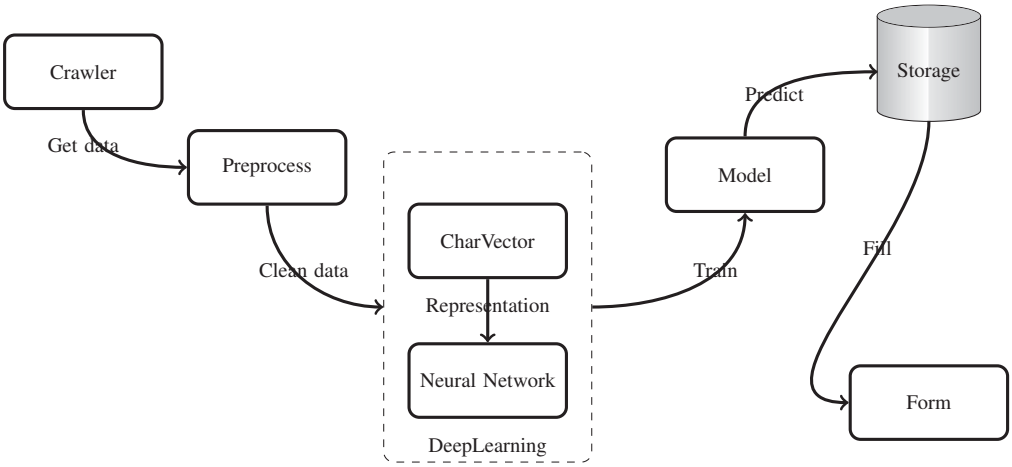

Fig. 1 System architecture

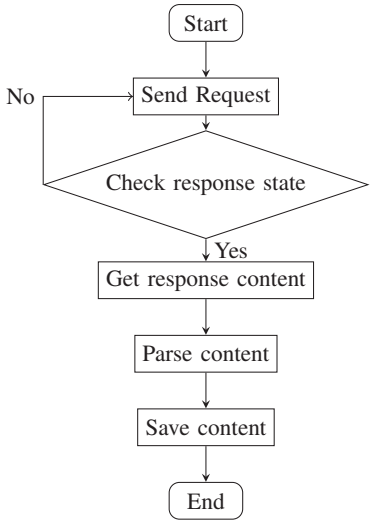

Fig. 2 Basic crawler process

Algorithm 1 Crawler algorithm

Input: Target page's url
Output: personal document

```
url=InitialUrl
while url ≠ null do
    response = requests.get(url)
    soup=BeautifulSoup(response.text, hxml)
    element = soup.select(DOM)
    for top ∈ element do
        for info ∈ top.select(label) do
            f.write(info. text)
    url = UrlQueue. pop()
```

text document types need to be crawled because these text documents are the data that we need to process.

B. Preprocessing

Data preprocessing is a very important step in neural networks because the data collected by the crawler have problems such as too much or too little data, too long or too short data. And the deviation between the amount of sampled data and the amount of real-world data may lead to meaningless metrics, so we need to delete some data to maintain the consistency between the sampled data and the real data. In particular, to improve the training speed, we delete the words that have no impact on training (collectively called stop words) according to Baidu's stop words table.

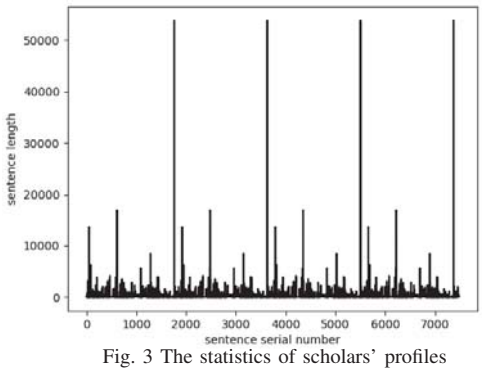

Fig. 3 The statistics of scholars' profiles

Fig. 3 shows the statistical result of scholars' profiles. The abscissa indicates the serial number of the sentence, and the ordinate indicates the length of the corresponding sentence. It is clear from the statistical graph that some data are longer or shorter than other data (collectively referred to as noise data), so deleting these data are beneficial to obtain better training results. After statistical analysis, choose to process the data with a larger deviation.

Algorithm 2 shows the denoising algorithm. In this algorithm, the input is noise data and threshold, and the output is clean data. First, we set thresholds *MaxLength* and *MinLength* to get valid data, and then open the file that store noise data and read it by line. Third, we delete data if data length is not between *MaxLength* and *MinLength*; otherwise, we delete stop word in these data.

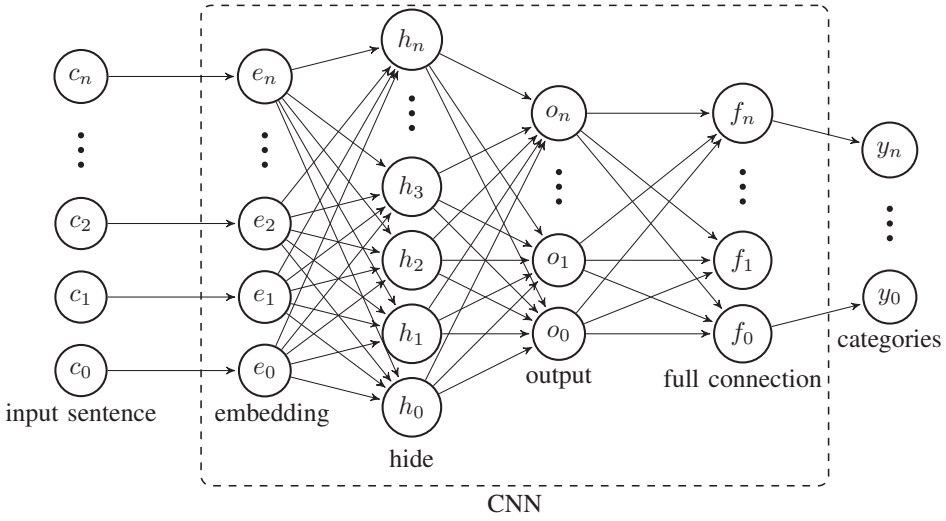

Fig. 4 Network model structure

Algorithm 2 Preprocessing algorithm

Input: Noise data and threshold **Output:** Clean data

```
MaxLength=m, MinLength = n
f= open('data.txt', 'r', 'utf8'):
lines = f.readlines()
for line ∈ lines do
    if class1 == line[:4] then
        if len(line)∉[MinLength,MaxLength] then
            del line
    else
        re.sub(stopwords, '', line)
```

C. Char Vectors

Since computers cannot directly process Chinese characters, the first step is to convert the Chinese into numbers. In addition to converting text to dictionary order, one-hot, and other methods, one can also use pre-trained word/char vectors. Many research institutions provide word/char vectors that have been trained. For example, word2vec[1], GloVe[2], elmo[3], BERT [4] are commonly used and training word/char vectors. The expression ability of the word vector is strong, and it is suitable for scenarios where the sample size is large enough. In contrast, the word vector has a better effect when the sample size is small [5]. The data size used in this paper is 1.28M, so we use the char vector.

Algorithm 3 shows how to load the pre-training char vector. In this algorithm, the pre-trained char vector, train data and the output are the char vector corresponding to the training data. First, we download the pre-trained char vector on the Internet. Second, we find the word vector corresponding to the training data in the pre-trained word vector, we use the UNK to represent char if char in train data, not in pre-trained data.

Algorithm 3 Loading the pre-training characters algorithm

Input: PretrainedCharVector, TraininngData
Output: CharVector

```
Download(PretrainedCharVector)
if char ∈ TrainData, ∉ PretrainedCharVector then
    char=UNK
else
    CharVector = PretrainedCharVector
```

D. Neural Network

Fig. 4 shows the network model structure. With the initial application of neural network models in the field of computer graphics, neural networks have also achieved good application prospects in the field of natural language processing in recent years. The two most popular neural network models are Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN). The data set used in this paper is labeled by itself and divided into six categories: personal profile, research area, education history, work experience, publication, and reward. This paper uses character-level CNN for text classification composed of inputs sequence, CNN, and probabilities of the classification result.

Fig. 5 shows the structure of the CNN used. The concept of CNN was first introduced by [6]. It can maximize the distinction between the input features. It consists of an input layer, hides layer, and outputs layer. And the hidden layer includes the input layer, convolutional layer, pooling layer, fully connected layer, and output layer. In this paper, we use a single-layer convolutional network to analyse the semantics of a text. The first layer is the embedding layer, which converts characters into char vectors and is the input of the CNN. The second layer is the convolution layer, input, and convolution kernel in the convolution layer for convolution operation. The third layer is the max-pooling layer, taking the maximum value of the feature vector obtained after each convolution operation. The fourth layer is the full-connection layer, where each node is connected to all the nodes in the previous layer and is used

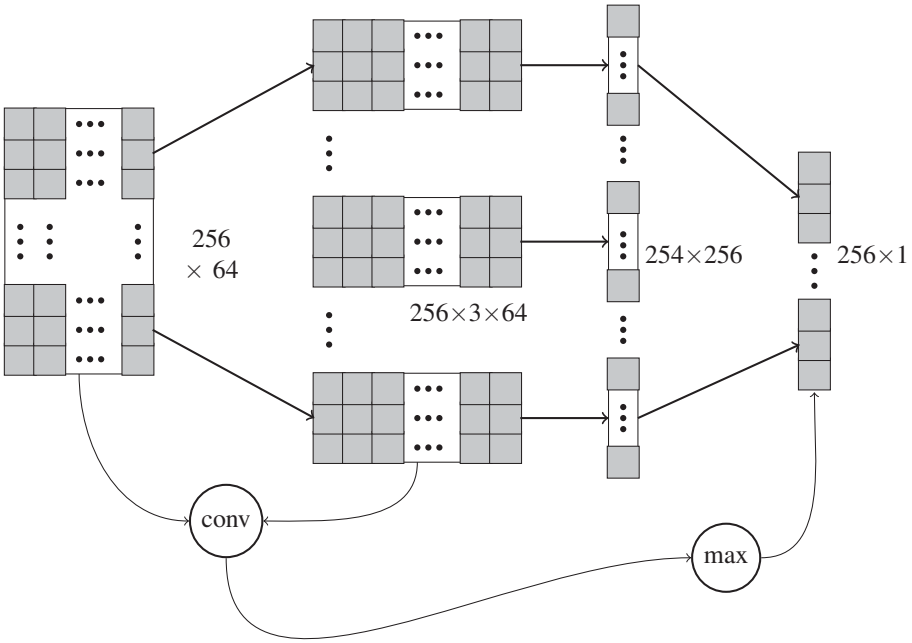

Fig. 5 Structure of CNN

Algorithm 4 CNN layer algorithm

Input: input sentence
Output: type

```
embedding=get_variable('embedding', [5000,64])
embedding_inputs=embedding_lookup( embedding, self.input_x)
conv.embedding_inputs=256×64
conv.filter = 256
conv.kernel_size=3
mp=max_pooling(conv)
fc=full_connection(mp, 128)
fc=dropout(fc, 0. 5)
fc=relu(fc)
logits = dense(fc, 6)
```

to synthesise the features extracted before. The fifth layer is the dropout layer, which is the same as the fully connected layer, the only difference is that only some nodes participate in the calculation. The sixth layer is activation layer, adding nonlinear factors to the network. The seventh layer gets a class probability.

Algorithm 4 shows the code of the convolution operation. First, we create a tensor of size 5000×64, and then use the embeddin_lookup function provided by TensorFlow to map the char vector to the input. The size of the input sentence matrix is 256×64, and there are 256 filters in total 3×64. The input matrix and filter are operated as follows:

$$\mathcal{C} = \mathcal{A} \cdot \mathcal{B}, \tag{1}$$

where $\mathcal{A}$ and $\mathcal{B}$ have the same order.

Each input is divided into 256 sub-matrices of size 3×64, and each sub-matrix and filter performed 256 operations, and finally 256 matrices of 254×1 are obtained. The max-pooling operation is as follows:

$$\mathcal{M} = \max \mathcal{C}. \tag{2}$$

It gets these matrices that hold input feature information (also called feature matrices). Next, in order to prevent overfitting, we perform a fully connected and dropout operation. Finally, we use Relu as the activation function to add the nonlinear network to the network, and convert the result after the Relu operation into the probability of each category.

E. Storage

There are more than a dozen kinds of commonly used data at present, such as relational databases (relational database model is to reduce complex data structures to simple binary relationships), key-value storage databases (key-value databases store data as keys). A collection of value pairs with keys as unique identifiers, document-oriented databases (such databases can store and retrieve documents in the formats of XML, JSON, and BSON). These documents are descriptive and present a hierarchical tree structure. They include mapping tables (collections and scalar values), graph databases (graph databases are used to store graph relational data), and other major categories. As a kind of document, tables are document-oriented databases. Document-oriented databases include MongoDB, CouchDB, Terrastore, RavenDB, and OrientDB. In this paper, we choose to use MongoDB because it can dynamically expand capacity as the load increases.

Python has a Pymongo driver to connect to MongoDB. We use the trained model to classify each piece of data and insert the trained data into the database. An automatically generated ID after inserting can be used for subsequent data

TABLE I
RESUME

Name		**Gender**		**Birthday**	
University			**Department**		
Research interest					
Position					
Education history					
Publicication					
Activity					

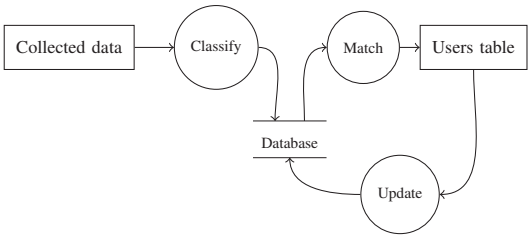

Fig. 6 System data flow diagram

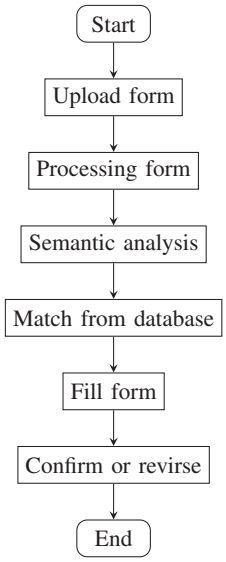

Fig. 7 Process of automatic filling

updates columns that do not currently exist but will exist in the database in the form of null and will not occupy memory. This data can be at any time. When auto-filling a form for a user, the system retrieves relevant data from the database according to The unique ID of the user.

F. Form Filling

Fig. 6 shows the data flow diagram of our system. Firstly, it uses the trained model to classify the data collected by the crawler and store the classified data in the database. When a user needs to fill a form, our system retrieves the data from the database. Afterwards, the user can manually change the data filled. After the change is made, the system will update the database accordingly. Therefore, the more forms the system fills for a user, the more data about the users the database contains.

Fig. 7 shows the process of filling the form. First, the user uploads the form to be filled into the system. Then the processing module extracts the entries from the form and analyse the word meaning of each entry to find the matching data from the database and fill it in the form. After the system completes the task of filling a form, the user needs to confirm the form.

III. AN EXAMPLE

This section shows how our system fills in a form by an example. The process of automatically filling the form is as follows:

Step 1: After the user is registered as a user of the system, after the registration is successful, the system will query the database for the corresponding data based on the school name and name filled in by the user. Whether the user's data are personal data. If so, bind the data-id to the user. If not, create a new empty data in the database.

Step 2: The user uploads the form to the system, the system analyses the form, extracts the entries, and then finds the corresponding entry through the synonyms table and the word similarity algorithm. If the entry exists, the corresponding content is returned to the user.

Step 3: After getting the returned result, the user needs to confirm whether or not it is correct. If not correct, the user can revise it. Then the revised content is updated in the database. For the content that does not exist in the database, users need to fill it in by themselves. After completing the filling, new data are added to the database.

Table I shows an example, which is a part of a resume (very common in daily office work). Specifically, the procedure of automatic filling the form is as follows:

1) The user uploads the form to the system.
2) Submit the form to the form processing module and extract the entries, including name, gender, date of birth, university, department, personal introduction, , education history.
3) Find synonymous items from the synonym table, such as 'personal profile' and 'personal information', 'self ntroduction', and 'personal introduction' express the same meaning.
4) According to the synonym table, find out the information corresponding to the user from the database. Table II shows some synonyms.
5) The query data are returned to the user, and then the user confirms and corrects. If no information is found in the database, the user is prompted to fill it out.

TABLE II
SYNONYMS

	synonymes 1	**synonymes** 2	**synonymes** 3	**synonymes** 4
1	Personal profile	Personal information	Self introduction	Personal introduction
2	Research interest	research direction	Research areas	Research objectives
3	Honor	Reward	Prize	Award
4	Education history	Education background	Education	
5	Published papers	Publication	Research Papers	Academic papers
6	Research achievements	Patent achievements	Academic achievements	Representative achievement

TABLE III
RESULTS FORM OF AUTOMATIC FILLING

Name		**Gender**		**Birthday**
University	Guangxi Normal University	**Department**	Computer Science	
Research interest	Fuzzy logic, Game theory, Automated negotiation, Smart software			
Position	British expert, doctoral supervisor, academic leader of key laboratory.			
Education history	1981-1985, undergraduate at Southwest University, China. 1987-1989, master student in Chinese Academy of Sciences, China. 1995-1998, PhD student at the University of New England, Austrailia.			
Publicication	1. Wenjun Ma, Yuncheng Jiang, Weiru Liu, Xudong Luo, Kevin McAreavey. Expected Utility with Relative Loss Reduction: A Unifying Decision Model for Resolving FourWell-Known Paradoxes. The Thirty-Second AAAI Conference on Artificial Intelligence. (Citations: 0) 2. Jieyu Zhan, Xudong Luo, Yuncheng Jiang. An Atanassov Intuitionistic Fuzzy Constraint Based Method for Offer Evaluation and Trade-off Making in Automated Negotiation. Knowledge-Based Systems, 2018, 139:170-188. (Citations: 4) . . .			
Activity	Participated in holding more than 100 academic conferences, serving as program committee member, senior member, co-chairman, and giving special invitation reports.			

TABLE IV
TEST SET EVALUATION

	precision	**recall**	**f1**
TextRNN	0. 85	0. 78	0. 82
TextRCNN	0. 91	0. 80	0. 85
FasText	0. 88	0. 92	0. 85
TextCNN	0. 96	0. 96	0. 96

6) After the user confirms that there are no errors, the user can download it to the local computer after saving. At the same time as the user downloads, the modified data of the user are also updated into the database accordingly.

Table III shows the results of filling the form. From the table, we can see that the system can fill the form accurately.

IV. EXPERIMENT

In our experiment, we use 2,100 data sets. We divide all the data into the training set, the test set, and the validation set according to the ratio of 6:2:2. In this section, we will carry out the experimental comparison and experimental evaluation.

A. Comparison

Table IV shows the results of different methods for classification tasks on this data set. From the table, we can see that CNN has achieved the best results on the data set in this paper, so we choose the method in this paper.

B. Evaluation

Figs. 8 and 9 show the accuracy curves of the training and the validation set as the number of training rounds increases during the training process. To observe the results more clearly, the number of training rounds is set to a larger number. The final accuracy rate on the training set is 99.55%, and the accuracy rate on the test set is 97.14%. So, we achieve good results have on both the training set and the test set.

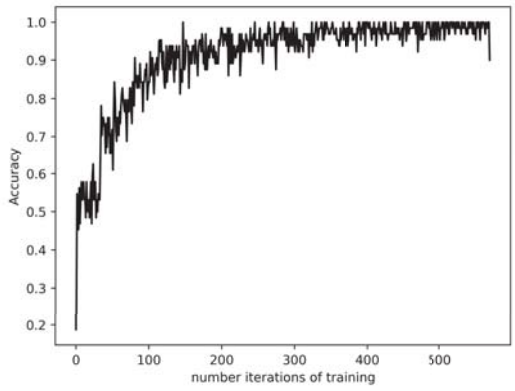

Fig. 8 Train sets

TABLE V
TEST SET EVALUATION

	precision	**recall**	**f1**
information	1.00	1.00	1.00
research interest	1.00	1.00	1.00
education history	0. 97	0. 99	0. 98
work experience	0. 97	0. 98	0. 97
paper	0. 96	0. 87	0. 91
achievement	0. 86	0. 90	0. 88

Table V shows the evaluation of the test set. On the test set, we use precision rate, recall rate, and f1 to test the training model. The calculation methods of the three criteria are as follows:

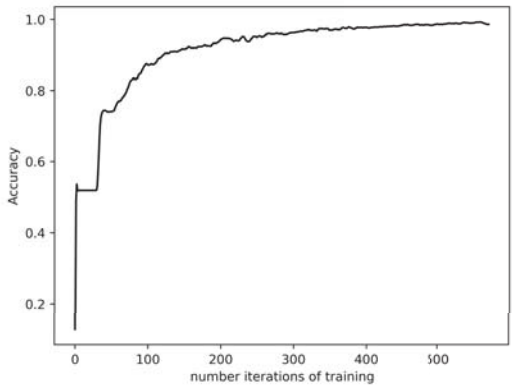

Fig. 9 Validation sets

$$\text{Recall} = \frac{\text{Related files retrieved by the system}}{\text{Total number of all related files in the system}},$$
$$\text{Precision} = \frac{\text{Relevant files retrieved by the system}}{\text{total number of files retrieved by the system}},$$
$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}.$$

As we can see from the table, the final model has the best test results in information and search fields, the worst test results in achievement, and good results in other categories, with the lowest f1 value of 0.91. So, it is reasonable to say that our model has achieved good results.

V. Related work

This section discusses related work to show how we advance the state of the art on this topic.

A. Crawler

Network data are abundant, but difficult and time consuming to download the data manually. Crawlers are undoubtedly a very good tool for automatically downloading data. Farag et al. [7] use an event model that can capture key event information, which crawl smart events and automatically track event keywords. Bal and Geetha [8] propose the client server architecture based smart distributed crawler for crawling web, which can use network bandwidth and storage capacity effectively. Gunawan et al. [9] use distributed focused crawler to improve the capabilities of data collection for article clustering. Deng et al. [10] optimise the web crawler system based on the scrapy framework,and improved the crawler speed. However, all of them are irrelevant to personal information of scholars.

B. Data Proprecessing

Data preprocessing is a very important part of neural network training, which directly affects the quality of the final model. Chandrasekar et al. [11] improve the prediction accuracy by data preprocessing. They use the supervised filter discretisation to construct a decision tree to get better classification model. Yang and Wang [12] did preprocessing and feature selection steps in the abnormal data classification, and get good results on KDD Cup99 dataset. Jiang and Xu [13] use the methods of data preprocessing in the intrusion detection system, improving the performance of the system. In this work, although we also use data preprocessing, but our data are about the academic information of scholars, while theirs are not.

C. Methods of Text Classification

There are a number of deep learning methods for classifying text. Joulin et al. [14] propose the FastText method. Although its effect is similar to other methods, its training speed is faster than other methods. Kim [15] propose the TextCNN method of using pretrained word vectors. Lai et al. [16] use TextRCNN method to obtain less noise. Yang et al. [17] use the TextRNN + Attention method to pay attention to important contents when constructing the document representation. Devlin et al. [4] use the BERT method to enable the static word vector to solve the problem of polysemy. Conneau et al. [18] use the VDCNN method to explore the effectiveness of deep models in text classification tasks. The optimal performance network in the text reaches 29 layers. Hassan and Mahmood [19] propose a deep learning model for text classification based on recurrent and convolutional layers. It uses bidirectional layers replace pooling layers in CNN in order to reduce the loss of local information, and to capture long-term dependencies of input sequences. Guo et al. [20] propose a CRAN model for text classification. It is a hybrid model that integrates the CNN and RNN effectively with the attention mechanism. Liu et al. [21] propose a ACNN model for text representation and classification. They use multiple attention mechanism to learn multiple context vectors, and finally use the Softmax classifier for text classification. Zheng and Zheng [22] propose a BRCAN model for text classification. The model uses bi-RNN to capture the long-term dependence among sentences and contextual information. Wang and Deng [23] propose the TCNN-SM model for text classification, which memory functional column retains of different granular memory and completes the selective storage of historical information. However, none of them use text classification methods for classifying scholars' personal information online.

D. Applications in Text Classification

Text classification has been applied into many ways. Moso et al. [24] use CNN with residuals to perform text classification of clinical data, which can help doctors easily browse a patients medical history. Silva et al. [25] use text classification methods to filter fake news. Zhong et al. [26] use CNN to classify and analyse the narrative of an accident in order to better understand the cause of the accident. Sriram et al. [27] use text classification to improve information filtering in order to prevent users from being overwhelmed by the raw data. Srivastava et al. [28] propose a healthcare text classification system to provide a better intelligent service for patients. Chen et al [29] use a Att-BiLSTM model in outpatient text classification system, which classify outpatient categories by textual content. Yang and Liu [30] use text classification

to judgement type of crimes. Sulea et al. [31] use text classification to predict the ruling of the French Supreme Court and the law area to which a case belongs to. Nevertheless, all the above studies do not use text classification for automatic filling a form for academic staff in their academic lives.

VI. Conclusion

To alleviate the workload of filling various forms in their academic lives so that they can work more efficient on their essential task, in this paper, we develop a system for automatically filling forms. Specifically, the system can automatically collect information of a scholar from the Internet, then use a trained classification model to classifying the collected data and save a database, and finally fill a form automatically.

In the future, our work in this paper could be extended in many ways. For example, it is impossible for our system to judge the correctness of the data collected online or to infer that there is no data in the database based on the existing data. This issue may be addressed by using personal Knowledge Graph. By establishing a personal knowledge graph, the user will eventually obtain more detailed and accurate information. From this information, user data can be automatically generated to maintain data consistency and also facilitate batch processing of various forms.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (Nos. 61762016 and 61662007) and Guangxi Key Lab of Multi-Source Information Mining & Security (No. 19-A-01-01).

References

[1] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Efficient estimation of word representations in vector space. In *Proceedings of the 2013 International Conference on Learning Representations*, pages 3111–3119, 2013.

[2] Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014.

[3] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2016 Confrence on North American Chapter of the Association for Computational Linguistics*, pages 2227–2237, 2018.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Confrence on Association for Computational Linguistics*, 2019.

[5] Xiaoya Li, Yuxian Meng, Xiaofei Sun, Qinghong Han, Arianna Yuan, and Jiwei Li. Is word segmentation necessary for deep learning of chinese representations? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3242–3252, 2019.

[6] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[7] Mohamed MG Farag, Sunshin Lee, and Edward A Fox. Focused crawler for events. *International Journal on Digital Libraries*, 19(1):3–19, 2018.

[8] Sawroop Kaur Bal and G Geetha. Smart distributed web crawler. In *Proceeding of the 2016 International Conference on Information Communication and Embedded Systems*, pages 1–5, 2016.

[9] Dani Gunawan, Amalia Amalia, and Atras Najwan. Improving data collection on article clustering by using distributed focused crawler. 2017.

[10] Deng Kaiying, Chen Senpeng, and Deng Jingwei. On optimisation of web crawler system on scrapy framework. *Proceeding of the 2020 International Journal of Wireless and Mobile Computing*, 18(4):332–338, 2020.

[11] Priyanga Chandrasekar, Kai Qian, Hossain Shahriar, and Prabir Bhattacharya. Improving the prediction accuracy of decision tree mining with data preprocessing. In *Proceeding of the 41st Annual Computer Software and Applications Conference*, volume 2, pages 481–484, 2017.

[12] Hongyu Yang and Fengyan Wang. Wireless network intrusion detection based on improved convolutional neural network. *Special Section On Security And Privacy In Emerging Decentralized Communication Environments*, 7:64366–64374, 2019.

[13] Shuai Jiang and Xiaolong Xu. Application and performance analysis of data preprocessing for intrusion detection system. In *Proceeding of the 2019 International Conference on Science of Cyber Security*, pages 163–177, 2019.

[14] Armand Joulin, Édouard Grave, Piotr Bojanowski, and Tomáš Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, 2017.

[15] Yoon Kim. Convolutional neural networks for sentence classification. In *Processing of the 19th Conference on Empirical Methods in Natural Language Processing*, page 17461751, 2014.

[16] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. Recurrent convolutional neural networks for text classification. In *Proceedings of the 29th AAAI Conference on Artificial intelligence*, pages 2267–2273, 2015.

[17] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, 2016.

[18] Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. Very deep convolutional networks for text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1107–1116, 2017.

[19] Abdalraouf Hassan and Ausif Mahmood. Efficient deep learning model for text classification based on recurrent and convolutional layers. In *Proceeding of the 16th IEEE international Conference on Machine Learning and Applications (ICMLA)*, pages 1108–1113, 2017.

[20] Long Guo, Dongxiang Zhang, Lei Wang, Han Wang, and Bin Cui. Cran: a hybrid cnn-rnn attention-based model for text classification. In *Proceeding of the 2018 International Conference on Conceptual Modeling*, pages 571–585, 2018.

[21] Tengfei Liu, Shuangyuan Yu, Baomin Xu, and Hongfeng Yin. Recurrent networks with attention and convolutional networks for sentence representation and classification. *Applied Intelligence*, 48(10):3797–3806, 2018.

[22] Jin Zheng and Limin Zheng. A hybrid bidirectional recurrent convolutional neural network attention-based model for text classification. *IEEE Access*, 7:106673–106685, 2019.

[23] Shiyao Wang and Zhidong Deng. Tightly-coupled convolutional neural network with spatial-temporal memory for text classification. In *Proceeding of the 2017 International Joint Conference on Neural Networks*, pages 2370–2376, 2017.

[24] Juliet Chebet Moso, Jonah Kenei, Elisha T Opiyo Omullo, Robert Oboko, et al. Deep cnn with residual connections and range normalization for clinical text classification. *Computer Science and information Technology*, 7(4):111–127, 2019.

[25] Renato M Silva, Roney LS Santos, Tiago A Almeida, and Thiago AS Pardo. Towards automatically filtering fake news in portuguese. *Expert Systems with Applications*, 146:113199, 2020.

[26] Botao Zhong, Xing Pan, Peter ED Love, Lieyun Ding, and Weili Fang. Deep learning and network analysis: Classifying and visualizing accident narratives in construction. *Automation in Construction*, 113:103089, 2020.

[27] Bharath Sriram, Dave Fuhry, Engin Demir, Hakan Ferhatosmanoglu, and Murat Demirbas. Short text classification in twitter to improve information filtering. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 841–842, 2010.

[28] Saurabh Kumar Srivastava, Sandeep Kumar Singh, and Jasjit S Suri. A healthcare text classification system and its performance evaluation: a source of better intelligence by characterizing healthcare text. In

Cognitive informatics, Computer Modelling, and Cognitive Science, pages 319–369. 2020.

[29] Che-Wen Chen, Shih-Pang Tseng, Ta-Wen Kuan, and Jhing-Fa Wang. Outpatient text classification using attention-based bidirectional lstm for robot-assisted servicing in hospital. *Information*, 11(2):106, 2020.

[30] Xi Yang and Ying Liu. Automatic extraction of theft judgment information in natural language. *Proceeding of the 18th International Conference on Electronic Business*, 2018.

[31] Octavia-Maria Sulea, Marcos Zampieri, Shervin Malmasi, Mihaela Vela, Liviu P Dinu, and Josef van Genabith. Exploring the use of text classification in the legal domain. *Analysis of information in Legal Texts*, 2017.

Shunzuo Wu is currently a master student at Guangxi Normal University, China.

Dr. Xudong Luo is currently a distinguished professor of Artificial Intelligence at Guangxi Normal University, China. He published one book and more than 160 papers including 2 in top journal *Artificial Intelligence*, one of which has been highly cited by, for example, MIT, Oxford, and CMU research groups. Prof. Luo has international recognised reputation: co-chair and (senior) members of PC of more than 100 international conferences or workshops, including major conferences IJCAI and AAMAS, and referees for many international journals such as top journal *Artificial Intelligence*. He is also invited to make a presentation of his work in more than 10 universities internationally, including Imperial College. His research focus is on the areas of agent-based computing, fuzzy sets and systems, decision theory, game theory, knowledge engineering, and natural language process. Prof. Luo has supervised or co-supervised more than 40 master students, Ph.D. students, and research fellows.

Yuanxiu Liao is currently a professor at Guangxi Normal University, China. Her research interests are mainly engaged on formal methods, artificial intelligence, urban computing, and big data analysis and processing.