Neural network technique with deep structure for improving.pdf

by

Submission date: 16-Sep-2022 04:54PM (UTC+0700)
Submission ID: 1901201186
File name: Neural network technique with deep structure for improving.pdf (881.87K)
Word count: 5724
Character count: 27976

TELKOMNIKA Telecommunication, Computing, Electronics and Control
Vol. 19, No. 4, August 2021, pp. 1208~1217
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v19i4.18878

Neural network technique with deep structure for improving author homonym and synonym classification in digital libraries

Firdaus[1], Siti Nurmaini[2], Varindo Ockta Keneddi Putra[3], Annisa Darmawahyuni[4], Reza Firsandaya Malik[5], Muhammad Naufal Rachmatullah[6], Andre Herviant Juliano[7], Tio Artha Nugraha[8]
[1, 2, 3, 4 ,6, 7, 8]Intelligent Systems Research Group, Universitas Sriwijaya, Palembang, Indonesia
[5]Communication Networks and Information Security Research Lab, Universitas Sriwijaya, Palembang, Indonesia

Article Info

Article history:

Received Nov 30, 2020
Revised Jan 3, 2021
Accepted Jan 20, 2021

Keywords:

Author name disambiguation
Bibliographic data
Deep neural networks
Homonym
Symonym

ABSTRACT

Author name disambiguation (AND), also recognized as name-identification, has long been seen as a challenging issue in bibliographic data. In other words, the same author may appear under separate names, synonyms, or distinct authors may have similar to those referred to as homonyms. Some previous research has proposed AND problem. To the best of our knowledge, no study discussed specifically synonym and homonym, whereas such cases are the core in AND topic. This paper presents the classification of non-homonym-synonym, homonym-synonym, synonym, and homonym cases by using the DBLP computer science bibliography dataset. Based on the DBLP raw data, the classification process is proposed by using deep neural networks (DNNs). In the classification process, the DBLP raw data divided into five features, including name, author, title, venue, and year. Twelve scenarios are designed with a different structure to validate and select the best model of DNNs. Furthermore, this paper is also compared DNNs with other classifiers, such as support vector machine (SVM) and decision tree. The results show DNNs outperform SVM and decision tree methods in all performance metrics. The DNNs performances with three hidden layers as the best model, achieve accuracy, sensitivity, specificity, precision, and F1-score are 98.85%, 95.95%, 99.26%, 94.80%, and 95.36%, respectively. In the future, DNNs are more performing with the automated feature representation in AND processing.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Siti Nurmaini
Intelligent Systems Research Group
Universitas Sriwijaya
Palembang 30137, Indonesia
Email: siti_nurmaini@unsri.c.id

1. INTRODUCTION

Author name disambiguation (AND) in the publication is a well-known problem. Such a technique used to overcome the problem of ambiguous in digital libraries (DL), including DBLP, Google Scholar, and others. When searching for an author's name or article title with a specific author's name on a DL, ambiguity problems often arise. Thus many related articles will appear with the same name or the same title in a bibliographic database [1]. Basically, AND condition occurs because of the following four reasons as follow [2]; (i) caused by someone who publishes with different names; (ii) many authors publish with the same name; (iii) incomplete data or errors; and (iv) the increasing number of articles and or journals published on DL. These four causes can be used as a reference for gathering all the information needed to find out and solve the problem of ambiguity.

To overcome the above problems, several researchers have proposed a solution [3]–[5]. However, the method can't guarantee accurate results. The AND topic was introduced by V.I. Torvik [6]. But specifically, the subject of AND is pursed in the case of homonyms and synonyms. The case of homonyms and synonyms is the core problem in AND which makes it even more complicated [7]. Homonyms are cases where two names are the same in a journal publication but are owned by different people, while Synonyms is the opposite case when there are different names but are owned by the same person [8], [9]. The research focused on the homonym case was conducted by Momeni F. using DBLP bibliographic data [10]. The study aims to evaluate the method used for the network co-authors in the case of homonym authors by clustering on the same name data (homonym). The research yielded good performance for the most names. Unfortunately, the method used still needed optimization. Many studies on AND topics pertain to the homonym and synonym cases [9], [11]-[13]. However, no specific research focuses on finding solutions in the cases of homonyms and synonyms classification.

Currently, there are several methods have proposed to give a solution in the AND classification problem based on machine learning with the supervised [14] and unsupervised approach [15]. One technique commonly used for the classification, and it has been proven to provide good results is support vector machine (SVM). Jason D. M. Rennie conducted a multiclass text classification using SVM, and it compared with Naïve Bayes [16]. The results prove that SVM can reduce losses 10% to 20% lower to Naïve Bayes, which means SVM has the performance to reduce losses to the lowest point. Jason D. M. Rennie present multiclass classification with SVM, a similar study was conducted by Giles M. Foody with image datasets [17]. In such a study SVM compares with discriminant analysis and decision trees. The results as shown by using SVM has 93.7% accuracy, discriminant analysis has 90% accuracy, and decision tree has 90.3% accuracy respectively.

One method of artificial neural networks with a "deep" structure known as the deep neural networks (DNNs) method is the most popular and widely used in classification problems. DNNs classifier produces an excellent performance for text processing [18]. In [19], the DNNs method significantly outperforms other methods and produces 99.31% accuracy in the Vietnamese author name. However, this method only detection author ambiguations, whereas, homonyms and synonyms are an essential problem in AND. To the best of our knowledge, only limited research about AND topic based on DNNs technique in the literature, and such research without investigation in homonym and synonym case. Hence, the homonym-synonym classification is desirable to the deep investigation. The rest of the paper is structured as follows. In the introduction, some related works to AND are discussed, and their capabilities and limitations are highlighted. The proposed method of DNNs describes in detail the working of homonym-synonym classification. We simulated the proposed algorithm on the Jinseok Kim dataset and compare it to baseline methods in experiments and discussions. In the end, we conclude with a discussion of conclusions and future work in conclusions.

2. MATERIAL AND METHODS

This paper proposes the text processing method to calculate appropriate features from the AND raw data. The method consists of data acquisition, data preparation, data pre-processing, feature reduction, classification, and validation, as in Figure 1. All the stages can be described as detail in the following section.

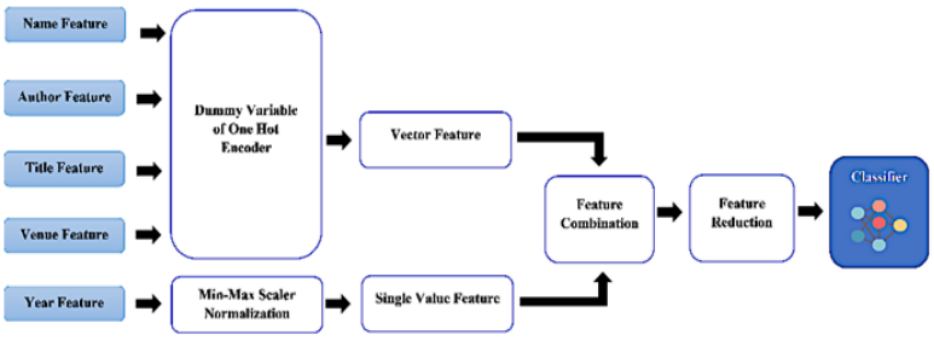


Figure 1. AND classification process

2.1. Data acquisition

The dataset used in this study came from Jinseok Kim [2], with raw data made by Giles Lee [20]. Initially, the raw data contained separate name data for one author in one file from many Digital Libraries (DL). The dataset has used by Richardo G. Cota [21] and Alan F. Santana [22]. Then, the raw data is improved by Muller [23] and he found that the dataset has a tendency towards DL DBLP and matches it with DBLP data. Furthermore, the dataset was cleaned up again and completed by Jinseok Kim [8] by adding missing data and eliminating unnecessary data based on DBLP data so that the dataset becomes a DBLP dataset with clear entities and attributes. In this dataset, there are six attributes always used because it has a great influence on AND data. The six attributes are a name, author, author ID, title, venue, and year. This dataset has the same length or amount of data for each attribute about 5018 data. There are 456 different names in the name attribute with three names over 100. The most names are Seong-Whan Lee about 125, David S. Johnson about 104, and Anoop Gupta about 104. While in the Authors attribute there are as many as 4654 names different, which shows that the name in the Authors attribute is very much different in number from the name in the Name attribute. This is due to the fact that in a journal publication, there are many authors who follow the author's name in the Name attribute. Whereas the Author ID attribute has 480 distinct data of Author ID data. It means there are 480 different labels scattered in the data along 5018.

2.2. Data preparation

All attributes contained in the dataset will be separated into feature attributes and label attributes. The feature attribute is commonly referred to as input, which is an attribute that will be used as input data to be processed by the classifier. The results will be grouped into one of the data in the label or output attribute. The label attribute is an attribute that is selected from many attributes that exist in the data that will be used as output (a place to group input data) from feature or input attributes. Feature attributes are taken from all the attributes that have important influences in the data aside from the label attributes. The label attribute is taken from the most specific and the unique attribute among all the attributes in the data that will be able to distinguish the data groups that are classified. In this study, the Author ID attribute is used as a label attribute because it is specific and unique.

In data preparation, the main task is to find individual homogeneous data in the dataset and store it as a new label (see in Figure 2). By adding a homonym label column to the dataset that can be initialized label 1 for the homonym and label 0 for the non-homonym. The next step is to find the synonym labels in the same way. The result is added as a new label column with initialization label 1 for synonym data and label 0 for non-synonym. Furthermore, two columns are added in the dataset, the first column of the homonym-synonym with label 1 labels and non-homonym-synonym with label 0. The second is a non-homonym-synonym label column, which is a label derived from data that is not included in the homonym label column and a synonym or the opposite of the synonym label column. Therefore, four new label columns are achieved, each new label column will be merged into a new label column with labels 0, 1, 2, and 3 representing each new label column homonym, synonym, homonym, non-synonym respectively. Thus, the training to be performed is a four-label training that results from the search for homonyms, synonyms, synonyms, and non-synonyms.

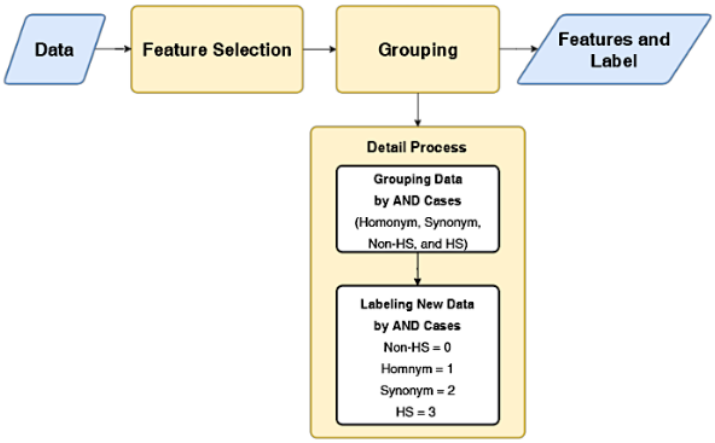


Figure 2. Data preparation process

2.3. Data pre-processing

In the pre-processing stage, all features except the Year feature are changed to dummy variable. The process of raw data processing into a dummy variable can be represented in Figure 3. These features can be easily interpreted by the classifier. For all features in the form of text, it will be changed into numeric form. For example, the Name feature is changed from name text to a number with a specific value or number in each name, and the value will be the same for the same name. The value is given in a random value. If there is a name like "Gupta", then the name will be represented as a random number from 0 to 9. That number will differ as much as the different names that exist. If there are 456 distinct names, then there will be 456 different numbers, which will represent each name in one column along 5018 data. After all, names become numbers and stored in a variable, a new variable will be prepared to store dummy variables whose contents are 0 and 1. The new variable will contain a number of columns the same number of the names. If there are 456 different names, then there will be 456 columns in the new variable. The length of the data on the new variable will be the same as the previous variable about 5018 data rows containing 0 and 1.

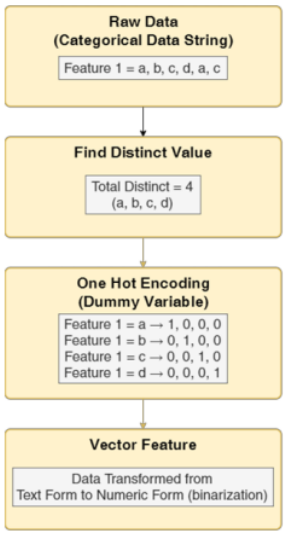


Figure 3. Data pre-processing

The number representing each name in the first variable will be used as the index column in the new variable. If the name "Gupta" is represented by the number 9, then in the ninth column in the new variable has the value 1 and the other column will be 0. Throughout the ninth column 9 from the first row to the 5018th row given a value of 1, if the previous variable contains the number 9. For example, if in the first variable the number 9 in the index row fifth and hundredth, then in the ninth column, the fifth and hundredth rows in the new variable will be worth 1 and otherwise 0. The rule will apply to all different numbers in the new variable. Pre-processing the author's feature is different from the Name feature, but it is still important to create a dummy variable. The step is to separate each text by name for all the names in the Authors feature and save it to a variable (e.g. variable "x"). Then a dictionary is also made which is stored in a variable (e.g. variable "z") from the name data in variable x. In the design of a dictionary does not change the shape or number of data in the "x" variable. Then all the names in the variable "x" are given the value 0, and also stored in a variable (e.g. variable "t"). The "x" variable is compared to the "z" variable by referring to the "t" variable, and the output of the process is a dummy variable as a feature to be an input classifier. Basically, the process changes the value of 0 in the variable "t" to 1, if the variable "z" finds the same name in the variable "x". Pre-processing of the Authors, Title and venue features are done in the same way because it has the same form of data. However, pre-processing in the year feature has a different step from the previous four features, because the Year feature is numeric, it only needs to be normalized into a value on a scale of 0 to 1. Finally, all features are collected into a single data with labels as the initial dataset. Then, the feature data is divided into 80% for training and 20% for testing.

2.4. Classification with deep neural networks

In this study, the multi-class-classification of AND based on DNNs use the implementation of the backpropagation (BP) algorithm for training and testing. The classifier is trained by using input data x and an annotated label as an output. The Softmax function is used as an activation function for the output layer of the classifier. By using the Softmax function, the output of each unit can be treated as the probability of each label. Here, let N be the number of units of the output layer, let x be the input, and let x_i be the output of unit i. Then, the output $p(i)$ of unit I is defined by in (1):

$$p(i) = \frac{e^{xi}}{\sum_{j=1}^{N} e^{xj}} \quad (1)$$

Cross entropy is used as the loss function of the classifier LF as follow:

$$L_F(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m} y_{ij} log(p_{ij}) \quad (2)$$

where n is the sample size, m is the number of classes, p_{ij} is the output of the classifier of class j of the i_{th} sample and y_{ij} is the annotated label of class j of the i_{th} sample.

In the experiment, all classes are divided into four categories, such as homonym, synonym, homonym-synonym, and non-homonym-synonym. Based on four classes, a DNNs comprises multiple nodes connected to each other, with each node representing the activation function. The experiment is conducted by increasing the number of hidden layers from layer 1 to layer 4, and all the performances are observed in order to choose the best model. The deep structures of NNs have 100 nodes in each layer. The activation function used in the input layer and at each hidden layer is ReLU, while the activation function used in the output layer is Softmax. In the DNN structure loss function is categorical cross-entropy with Adam optimizer. All experiments are conducted with a tuning learning rate from 0.0001 decreases to 0.1, with a batch size increase from 8 to 64 with 50 epochs. The parameter fixes for each experiment starting with 1 hidden-layer with batch size 8 up to 4 hidden-layers and batch size 64. The proposed DNNs structure can be seen in Figure 4.

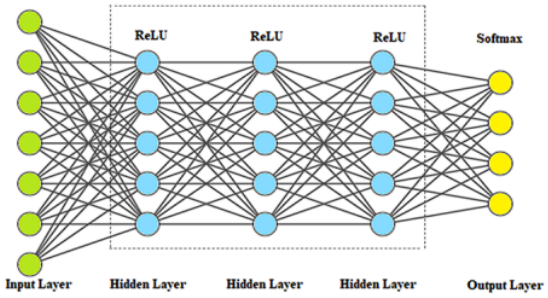


Figure 4. DNNs structure

2.5. Model evaluation

In the evaluation stage of the model that has been built, it is looking for the value of predicting testing features and testing labels of the model built, then utilizing the predicted value obtained to obtain the confusion matrix of the model. The classifier must be produced the trust value of the prediction results. The classifier makes correct predictions even though it is roughly interpreted as a probability. However, the possibility of getting the correct prediction is not enough to only give one number. The five measures of one beat H are as shown in (3) to (7) as follows:

$$sensitivity\ (Sen) = \frac{\sum_{H=1}^{K} TP_H}{\sum_{H=1}^{K} TP_H + \sum_{H=1}^{K} FN_H} \quad (3)$$

$$spesificity\ (Spe) = \frac{\sum_{H=1}^{K} TN_H}{\sum_{H=1}^{K} TN_H + \sum_{H=1}^{K} FP_H} \quad (4)$$

$$accuracy\ (Acc) = \frac{\sum_{H=1}^{K} TP_H + \sum_{H=1}^{K} TN_H}{\sum_{H=1}^{K} TP_H + \sum_{H=1}^{K} TN_H + \sum_{H=1}^{K} FP_H + \sum_{H=1}^{K} FN_H} \quad (5)$$

K is the number of beat types. TP_H (true positives) is the number of H types that are correctly classified. TN_H (true negative) is the number of not-H types that are correctly classified. FP_H (false positive) is the number of not-H types that are incorrectly predicted as H types. FN_H (false negative) is the number of H types that are incorrectly predicted as not-H types.

3. RESULTS AND DISCUSSION

The four classes of AND are classified with DNNs. The classes consist of non-homonym-synonym (class 0), homonym (class 1), synonym (class 2), and homonym-synonym (class 3). The twelve models are fine-tuned with the different batch sizes and hidden layers in each four class. One hidden layer is implemented in the first to fourth model. The fourth to eight model uses two hidden layers. Last, the ninth to twelfth model uses three hidden layers. Each hidden layer consists of a batch size of 8, 16, 32, and 64, respectively. For the non-homonym-synonym represented in Table 1, overall, the good performance with the percentage up 95% have worked out in twelve models of DNNs. The best model can be seen in the seventh and eighth model, which use two hidden layers and batch sizes 32 and 64, respectively. The differences between these two models are not significant.

Table 1. DNNs performance for non-homonym-synonym classification (class 0)

Model	Performance Evaluation (%)					Number of Data
	Accuracy	Sensitivity	Specificity	Precision	F1-Score	
1	98	97	98	99	98	615
2	96	95	98	99	97	600
3	97	96	99	99	98	590
4	97	95	99	99	97	599
5	98	97	99	99	98	600
6	97	97	98	98	96	588
7	98	97	99	99	98	599
8	98	97	99	99	98	606
9	97	96	99	99	97	590
10	97	96	98	99	97	609
11	97	96	98	98	97	611
12	97	96	99	99	98	584

For the homonym class, Table 2 shows the results of up to 87% as overall in the twelve models. The number of homonym data smaller than the non-homonym-synonym class. The fifth model with two hidden layers and a batch size of 8 is the best model for this class. For the Synonym class in Table 3, the results achieve the performance up to 92%, moreover, in the second (one hidden layer) and eleventh model (three hidden layers), the sensitivity gets the perfect performance, 100%. Different from three classes, the last class of homonym-synonym, the results are not good enough, with the percentage up 68%. From Table 4, the performance in sensitivity, precision, and F1-score are not satisfying. Adding more hidden layers not give significant results, because it does not really affect the DNNs performance. The limited parameters that fine-tuned (layers and batch size) are not always given a promising result. The learning rate and optimizer in each model are highly to consider. The proposed DNNs architecture in each class cannot be equated due to the proportion of the number of data is different.

Table 2. DNNs performance for homonym classification (class 1)

Model	Performance Evaluation (%)					Number of data
	Accuracy	Sensitivity	Specificity	Precision	F1-Score	
1	97	92	98	93	92	147
2	97	93	98	92	92	154
3	97	96	98	89	92	153
4	97	97	97	86	91	153
5	98	98	99	94	96	159
6	98	92	99	95	94	164
7	98	95	98	93	94	151
8	98	97	98	94	95	157
9	97	92	98	89	90	158
10	97	94	98	90	92	165
11	97	93	97	87	90	158
12	97	97	98	89	93	159

Table 3. DNNs performance for synonym classification (class 2)

Model	Performance Evaluation (%)					Number of data
	Accuracy	Sensitivity	Specificity	Precision	F1-Score	
1	98	99	98	95	97	212
2	98	100	98	92	96	218
3	99	99	99	97	98	232
4	98	99	98	94	96	218
5	98	99	98	93	96	215
6	99	99	99	96	97	210
7	98	98	98	96	97	218
8	99	99	99	96	97	204
9	98	99	98	93	96	219
10	98	97	98	93	95	196
11	98	100	98	94	97	209
12	98	99	98	94	97	227

Table 4. DNNs performance for homonym-synonym classification (class 3)

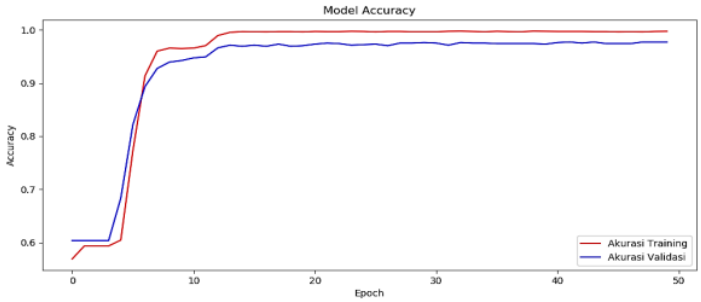
Model	Performance Evaluation (%)					Number of data
	Accuracy	Sensitivity	Specificity	Precision	F1-Score	
1	98	80	99	70	75	212
2	98	88	99	75	81	218
3	98	75	99	72	73	232
4	98	68	99	76	72	218
5	99	81	99	90	85	215
6	98	90	98	71	80	210
7	98	83	98	72	77	218
8	99	89	99	89	89	204
9	98	80	99	75	77	219
10	98	86	99	76	81	196
11	98	70	99	84	77	209
12	98	77	99	91	83	227

In this study, the training data use 80% of the data, and the remaining data percentage is used for the testing process. The twelve models that implemented in four class of AND category use the same hyperparame11 with the learning rate of 0.0001. Further, the activation function in the hidden layer is ReLU and Softmax in the output layer. The number of nodes is 100 in the input layer. The total epochs for the twelve models are 50. The comparison of DNNs architecture in four classes can be presented in Table 5. From Table 5, the eighth model showed accuracy, sensitivity, specificity, precision, and F1-Score, which is 98.85%, 95.95%, 99.26%, 94.80%, and 95.36%, respectively. Overall, the eighth model shows the best results in other models. The proposed DNNs architecture is two hidden layers with a batch size of 64.

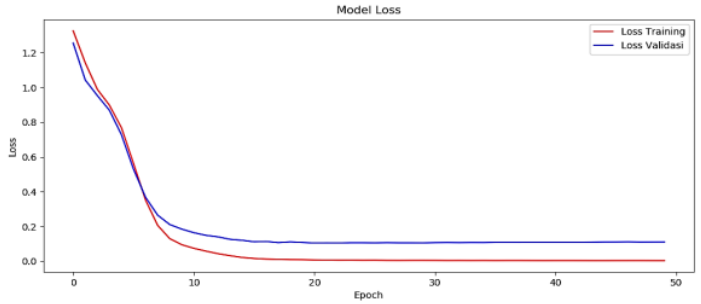
Table 5. Average performance for 4 classes

Model	Performance Evaluation (%)				
	Accuracy	Sensitivity	Specificity	Precision	F1-Score
1	98.36	92.74	98.86	89.59	90.92
2	98.01	94.49	98.68	89.80	91.96
3	98.41	91.98	99.01	89.76	90.82
4	97.81	90.17	98.65	89.11	89.47
5	98.71	94.11	99.15	94.65	94.30
6	98.36	94.94	98.86	90.58	92.46
7	98.46	93.90	99.01	90.36	92.01
8	98.85	95.95	99.26	94.80	95.36
9	97.81	91.97	98.61	89.39	90.62
10	97.96	93.67	98.59	90.07	91.77
11	97.96	90.30	98.54	91.55	90.73
12	98.31	92.74	98.92	93.71	93.03

To verify the proposed DNNs model, the accuracy and the loss curve are shown in Figures 5 (a) and (b), respectively. In Figure 5 (a), the accuracy of the training and testing set starts from around 60% in the first epoch. Later, the accuracy is increased with more epochs. Then, in the above 10 epochs, the accuracy gets the perfect score in the training set. The littl1gap between accuracy in training and testing set lead the overfitting. Figure 5 (b) shows decreasing error along with the increasing epochs in the training and testing set. Both curves produce good shapes in the processing with DNN.



(a)



(b)

Figure 5. Accuracy and loss curve of proposed dnns architecture: (a) accuracy curve, (b) loss curve

In this study, the benchmarking study of DNNs with other ML methods is proposed. The other ML classifiers such as SVM (matrix kernel polynomial, RF, and linear), decision tree (CART and C4.5 algorithm), and conventional artificial neural networks (ANNs) is compared to the proposed DNNs architecture with three hidden layers. The result of classifiers can be seen in Table 6, in both training and testing set.

Tables 6 and 7 show the accuracy of the neural networks, whether in DNNs or ANNs, outperform SVM and decision tree in all five metrics. It shows the neural network still be the promising classifier for AND case, due to neural network much more tedious than using an off-the-shelf classifier like SVM. Table 7 shows the same accuracy of DNNs and ANNs is 98.85% in the testing set. However, for the greater and a large amount of data in the digital library, making DNNs more have the prospect of being the suggested classifier than ANNs in the future. DNNs are more performing with the combination of automated feature representation in Big Data.

Table 6. Accuracy benchmarking with other ML methods

Methods	Training Accuracy	Testing Accuracy
DNNs (3 layers)	99.73	98.85
SVM with matrix kernel polynomial	99.85	98.56
SVM with matrix Kernel RBF	99.85	98.61
SVM with Matrix Kernel Linear	99.82	91.88
Decision tree with CART algorithm	99.80	90.29
Decision tree with C4.5 algorithm	99.80	88.90
ANNs	99.68	98.85

Table 7. Performance benchmarking with other ML methods

Methods	Performance Evaluation (%)				
	Accuracy	Sensitivity	Specificity	Precision	F1-Score
DNNs	98.85	95.95	99.26	94.80	95.36
SVM	98.61	95.08	99.18	89.44	91.80
Decision tree	90.29	74.34	92.11	67.99	70.64
ANNs	98.80	95.60	99.26	94.61	95.06

The case of AND is not a new study in bibliographic data. This paper also compares the proposed DNNs with some previous literature. Table 8 shows the benchmarking study with conventional and neural network methods. Tran *et al*. [19] proposed DNNs for author disambiguation case. The DNNs method achieves 99.31% in terms of accuracy. DNNs are significantly outperformed other methods that use a predefined feature set. However, homonyms and synonyms are essential problems in AND. Still, Chin *et al*. [24] implement Microsoft Academic Search (MAS) for string comparison in the author's name. The method is merging the results of two predictions further boost the performance. This approach method achieves F1-score 0.99202 on the private leader board, while 0.99195 on the public leader board. Then, Wang *et al*. [25] use Publication, Web Page & News Storie's dataset with a pairwise factor graph (PFG) and interaction of user to enhance the performance. The F1-score achieves 0.815 in CALO dataset case. The last, with our proposed DNNs method with three hidden layers, DNNs can show 98.85% of accuracy in case of non-homonym-synonym, homonym-synonym, synonym, and homonym authors.

Table 8. Performance benchmarking with previous methods

Authors	Method	Dataset	Case	Accuracy	F1-Score
Tran *et al*. [19]	DNNs	Vietnamese author-name	Author ambiguations	99.31%	-
Chin *et al*. [24]	Merging results of two predictions further boost the performance	Microsoft Academic Search (MAS)	String Comparison	-	0.99
Wang *et al*. [25]	A pairwise factor graph (PFG) model	Publication, Web Page & News Stories	CALO Dataset	-	0.82
Proposed	DNNs	DBLP	Non-Homonym-Synonym, Homonym-Synonym, Synonym, and Homonym Authors	98.85%	0.95

4. CONCLUSION

The main challenge to AND is that the data is becoming more and more complex and dynamic. It requires the AND algorithm to be extensible and flexible for different scenarios. This paper proposes one of ML algorithms, DNNs, for multiclass classification in non-homonym-synonym, homonym-synonym, synonym, and homonym authors task. For the classification task, a knowledge base is required. The DBLP computer science bibliography dataset with five attributes have implemented for the AND problems. Experimental results show that on all the twelve different models, our proposed method outperforms the common existing ML methods, such as SVM and decision tree. The DNNs performance with three hidden layers as the best model, achieve accuracy, sensitivity, specificity, precision, and F1-score are 98.85%, 95.95%, 99.26%, 94.80%, and 95.36%, respectively. In the future, DNNs are more performing with the automated feature representation in AND processing.

REFERENCES

[1] K. Canese and S. Weis, "PubMed: the bibliographic database," *National Library of Medicine*, 2013.
[2] J. Kim and J. Kim, "The impact of imbalanced training data on machine learning for author name disambiguation," *Scientometrics*, vol. 117, no. 1, pp. 511-526, Aug. 2018, doi: 10.1007/s11192-018-2865-9.
[3] C. A. D. Angelo, C. Giuffrida, G. Abramo, and C. A. D. Angelo, "A heuristic approach to author name disambiguation in bibliometrics databases for large-scale research assessments," *J. Am. Soc. Inf. Sci. Technol.*, vol. 62, no. 2, pp. 257-269, 2011, doi: 10.1002/asi.21460.
[4] K.-H. Yang, H.-T. Peng, J.-Y. Jiang, H.-M. Lee, and J.-M. Ho, "Author name disambiguation for citations using topic and web correlation," *Res. Adv. Technol. Digit. Libr.*, pp. 185–196, Sep. 2008, doi: 10.1007/978-3-540-87599-4_19.
[5] A. A. Ferreira, A. Veloso, M. A. Gonçalves, and A. H. F. Laender, "Effective self-training author name disambiguation in scholarly digital libraries," *Proceedings of the 10th annual joint conference on Digital libraries*, 2010, pp. 39-48, doi: 10.1145/1816123.1816130.
[6] V. I. T. and N. R. Smalheiser, "NIH Public Access," *US National Library of Medicine*, vol. 3, no. 3, pp. 1–30, 2010.
[7] M. R. Ackermann and F. Reitz, "Homonym detection in curated bibliographies: learning from dblp's experience," *International Conference on Theory and Practice of Digital Libraries*, Jun. 2018, pp. 59-65.
[8] J. Kim, "Evaluating author name disambiguation for digital libraries : a case of DBLP," *Scientometrics*, vol. 116, no. 3, pp. 1867-1886, 2018.
[9] F. Momeni and P. Mayr, "Using co-authorship networks for author name disambiguation," *Proc. 16th ACM/IEEE-CS Jt. Conf. Digit. Libr.-JCDL '16*, pp. 261-262, Jun. 2016.

[10] F. Momeni and P. Mayr, "Evaluating co-authorship networks in author name disambiguation for common names," *International Conference on Theory and Practice of Digital Libraries*, 2016, pp. 386-391.
[11] J. Kim and J. Diesner, "Distortive effects of initial-based name disambiguation on measurements of large-scale coauthorship networks," *J. Assoc. Inf. Sci. Technol.*, vol. 67, no. 6, pp. 1446-1461, 2016, doi: 10.1002/asi.23489.
[12] S. Heeffer, B. Thijs, and W. Glänzel, "Are registered authors more productive?," *Proc. ISSI 2013*, vol. 2, pp. 1864-1867, January 2013.
[13] D. E. Smith, J. Frank, and W. Cushing, "The ANML language," *The ICAPS-08 Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*, 2008.
[14] S. Goldman and Y. Zhou, "Enhancing supervised learning with unlabeled data," *ICML*, 2000, pp. 327-334.
[15] Q. V Le, "Building high-level features using large scale unsupervised learning," *2013 IEEE international conference on acoustics, speech and signal processing*, Dec. 2013, pp. 8595-8598.
[16] J. D. M. Rennie and R. Rifkin, "Improving multiclass text classification with the support vector machine," Massachusetts Institute of Technology as AI Memo 2001-026 and CBCL Memo 210, Oct. 2001. [Online]. Available: http://hdl.handle.net/1721.1/7241
[17] G. M. Foody and A. Mathur, "A relative evaluation of multiclass image classification by support vector machines," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 6, pp. 1335-1343, Jun. 2004, doi: 10.1109/TGRS.2004.827257.
[18] W. Liu, *et al.*, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11-26, Apr. 2017, doi: 10.1016/j.neucom.2016.12.038.
[19] H. N. Tran, T. Huynh, and T. Do, "Author name disambiguation by using deep neural network," *Asian Conference on Intelligent Information and Database Systems*, Apr. 2014, pp. 123-132.
[20] H. Han, L. Giles, H. Zha, C. Li, and K. Tsioutsiouliklis, "Two supervised learning approaches for name disambiguation in author citations," *Proceedings of the 2004 Joint ACM/IEEE Conference on Digital Libraries*, 2004, pp. 296-305.
[21] C. S. and G. Marchionini, "Awareness in CIS," *Int. Rev. Res. Open Distance Learn*, vol. 14, no. 4, pp. 90–103, 2013.
[22] A. F. Santana, M. A. Gonçalves, A. H. F. Laender, and A. A. Ferreira, "On the combination of domain-specific heuristics for author name disambiguation: the nearest cluster method," *Int. J. Digit. Libr.*, vol. 16, no. 3, pp. 229-246, 2015.
[23] M. C. Müller, F. Reitz, and N. Roy, "Data sets for author name disambiguation: an empirical analysis and a new resource," *Scientometrics*, vol. 111, no. 3, pp. 1467-1500, Mar. 2017.
[24] W.-S. Chin, Y. Chin Juan, Y. Zhuang, F. Wu, "Effective string processing and matching for author disambiguation," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3037-3064, Aug. 2014, doi: 10.1145/2517288.2517295.
[25] X. Wang, J. Tang, H. Cheng, and S. Y. Philip, "Adana: Active name disambiguation," *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, Dec. 2011, pp. 794-803, doi: 10.1109/ICDM.2011.19.

Neural network technique with deep structure for improving.pdf

ORIGINALITY REPORT

13%	11%	9%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.mdpi.com Internet Source	3%
2	www.kdd.org Internet Source	2%
3	dx.doi.org Internet Source	1%
4	explore.openaire.eu Internet Source	1%
5	rie.binadarma.ac.id Internet Source	1%
6	Ijaz Hussain, Sohail Asghar. "LUCID: Author name disambiguation using graph Structural Clustering", 2017 Intelligent Systems Conference (IntelliSys), 2017 Publication	1%
7	Submitted to De Montfort University Student Paper	1%
8	ijai.iaescore.com Internet Source	

		1 %
9	Xuezhi Wang. "ADANA: Active Name Disambiguation", 2011 IEEE 11th International Conference on Data Mining, 12/2011 Publication	1 %
10	citeseerx.ist.psu.edu Internet Source	1 %
11	Lecture Notes in Computer Science, 2014. Publication	1 %

Exclude quotes	On	Exclude matches	< 1%
Exclude bibliography	On		