Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 95 (2016) 153 – 158

Procedia Computer Science

Complex Adaptive Systems, Publication 6
Cihan H. Dagli, Editor in Chief
Conference Organized by Missouri University of Science and Technology
2016 - Los Angeles, CA

Entity Resolution Using Convolutional Neural Network

Ram Deepak Gottapu[a], Dr. Cihan Dagli[a], Dr. Bharami Ali[a]*

Missouri University of Science and Technology, Rolla, MO, 65409, USA

Abstract

Entity resolution is an important application in field of data cleaning. Standard approaches like deterministic methods and probabilistic methods are generally used for this purpose. Many new approaches using single layer perceptron, crowdsourcing etc. are developed to improve the efficiency and also to reduce the time of entity resolution. The approaches used for this purpose also depend on the type of dataset, labeled or unlabeled. This paper presents a new method for labeled data which uses single layered convolutional neural network to perform entity resolution. It also describes how crowdsourcing can be used with the output of the convolutional neural network to further improve the accuracy of the approach while minimizing the cost of crowdsourcing. The paper also discusses the data pre-processing steps used for training the convolutional neural network. Finally it describes the airplane sensor dataset which is used for demonstration of this approach and then shows the experimental results achieved using convolutional neural network.

© 2016 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Peer-review under responsibility of scientific committee of Missouri University of Science and Technology

Keywords: word stemming; word embedding; convolutional neural network; crowdsourcing, hybrid machine-human model

* Corresponding author. Tel.: +0-000-000-0000 ; fax: +0-000-000-0000 .
E-mail address: rgrk6@mst.edu

1877-0509 © 2016 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
Peer-review under responsibility of scientific committee of Missouri University of Science and Technology
doi:10.1016/j.procs.2016.09.306

1. Introduction

Entity resolution or record linkage is the task of identifying records that refer to same entity in a dataset or across multiple datasets. It becomes especially important when merging data from different sources. For example if we consider the product descriptions posted on a certain e-commerce website and the company wanted to classify these records based on the product model using the product descriptions as shown below:

Table 1. Product Dataset

Record	Product Description	Product Label
r_1	iPhone 5 16-white	Iphone 2015
r_2	iPhone 5th generation 16GB WiFi White	Iphone 2015
r_3	Samsung Galaxy Tab E 9.6" with 16 GB and Wifi (Black)	Galaxy 2015
r_4	Samsung Galaxy Tab E 9.6 inches 16 -Black	Galaxy 2015
r_5	Apple iPhone 5 16GB WiFi White	Iphone 2015

From the table we can infer that records r1, r2 and r5 refer to one product label/entity and r3 & r4 refer to another product label/entity. Since our approach deals with labeled data, we can consider the product label/entities as labels. However, the descriptions are not completely identical and hence we cannot use simple string comparisons to identify which records are identical and link them to product label. Probabilistic method is the most common algorithm used for entity resolution, as set forth by Howard Newcombe et al.[1, 2] , and formalized by Fillegi and Sunter [3]. However, it is not effective for above example as the method requires dataset to have multiple columns of data to find records that have high probability of being similar [4]. [5] described a hybrid machine-human approach which can be applied to datasets having single columns of sentences as show in above table. It uses Jaccard similarity (machine part) and crowdsourcing (human part) to perform entity resolution. In hybrid machine-human model the machine part is usually an algorithm and the human part is usually a task performed by the humans. For cases such as entity resolution, the algorithm part performs certain amount of record links and those linked records are then uploaded as small tasks on the crowdsourcing platform. These tasks are solved by people who get paid for each task they complete. The tasks are usually referred to as HIT's (human intelligence tasks) and the people working on those tasks are called crowd [5].

This is a very efficient way to perform entity resolution but using crowd for such purpose costs money. If there are millions of records on which entity resolution has to be performed, then the obvious goal is to reduce the tasks given to the crowd. This can be achieved if the machine part successfully classifies majority of the dataset. Similarity functions such as Jaccard similarity are not efficient for such a task as it requires generating record pairs in order to find which records are identical. For example if we have n records then we have to generate $n(n-1)/2$ record pairs and perform comparisons using similarity function in order to find out which records have high similarity. In addition to that, if the records are too distinct, similarity functions will identify those records as dis-similar records.

For example, Jaccard similarity over two records is defined as the size of the set intersection divided by the size of the set union.

$J(r_5, r_2) = (Iphone, 16GB, white, WiFi)/(Iphone, 16GB, white, Apple, 5th, 5, generation, WiFi) = 0.5$

Similarly $J(r_1, r_2) = 0.25$ and $J(r_3, r_4) = 0.59$. The records whose similarity is above a certain threshold are converted into tasks and uploaded in the crowdsourcing platform [5]. The similarity value is usually chosen above 0.5 so that the records having more than 50% similarity are sent to the crowd. However, we can observe cases like $J(r_1, r_2)$ whose similarity is less than threshold and yet they belong to same product label. The reason is that the words used are completely different to describe the same product. Hence we need a model that can assign correct labels even when the words used to describe the product are different and also which can reduce the tasks that have to be given to the crowd. In this paper, we also use hybrid machine-human approach but the machine part uses

convolutional neural network to reduce the number of tasks given to the crowd and also to improve the efficiency of predicting the right label.

The convolutional neural networks eliminates the need for the generation of record pairs and instead gives probabilities for each label. Once the entire dataset gets a label (highest probability) using the trained convolutional neural network, it implies that all the records under each entity/label are identical records. We can now create HIT's using these records and show that the number of tasks created using this approach is less than those created using Jaccard similarity function.

Section 2 describes the pre-processing steps that have to be used before training the CNN. Section 3 describes the CNN training and section 4 gives a brief discussion about crowdsourcing. Finally, Section 5 describes the experimental results and section 6 gives the conclusion.

2. Pre-processing methodology

Since all neural network algorithms including convolutional neural networks work only with numerical data, there is a need to represent the given data in a numerical format to learn the complex features present in the data. In addition to that, if we use words that uniquely represent each record, the accuracy of the CNN can be improved. For this purpose, keywords generation and word embedding are discussed in the following sections. It is important to note that keyword generation may not be possible for all types of datasets. If the dataset has records which needs to use all the words, then there is no need to generate keywords.

2.1. Keyword generation

The record r_3, shown in Table 1, contains some words that do not provide any significant contribution to identify the product label. For example, the words in the product description like "with" and "and" do not give any contribution to the training of CNN as they are not unique to the product label. Also, in some cases it may reduce the efficiency of the network. Hence such words can be removed and only the words which represent the product label uniquely can be used for the training of CNN. If the words in the dataset cannot be excluded, then we can treat all the words as keywords.

In addition to keyword generation, we also need to remove special characters from the dataset like apostrophes, brackets, commas, full stops, underscores etc.

2.2. Creating look-up table

Using all the words in the dataset that will be used for training, dictionary D is created. Each word $w_i \in D$ will be given an index $I \in \mathrm{R}$ which uniquely represents the word. These words along with their indices act as a look-up table $LT(.)$. Hence, whenever a record from the dataset is given as input to the CNN, corresponding indices for the words in the record are used as follows:

$$LT(w_i) = I_i \text{ where } I_i \text{ is the index of } i^{th} \text{ word.} \quad (1)$$

2.3. Handling variable record length

The look up table maps the records to its indices. However the size of each record is not fixed and this causes varying size inputs. Unfortunately CNN cannot be trained with a varying size input. The simplest solution is to choose an upper limit for the sentence length and apply padding. If n is the maximum length that any record in the dataset has, then the input is as follows:

$$I_1 \oplus I_2 \oplus I_3 \oplus I_4 \oplus 0 \oplus 0 \dots \dots n \text{ where } \oplus \text{ represents concatenation} \quad (2)$$

2.4. Word embedding

Word embedding is an important step before training the CNN. Indices alone do not have any features that can be learned. In order to train the CNN, each word/index must have a unique representation so that the network can uniquely identify the word. The word embedding is an approach which represents each word in a high dimensional vector. If the dimensionality of each word/index is d, then each word/ index can be represented as:

$$I_n = (x_1, x_2, x_3, x_4, \ldots . x_d) \tag{3}$$

The index is connected to I_n by weights through a fully connected network as follows:

$$index\ to\ embedding\ layer\ output(I) = lin\left(W^T_{embedding} I + b_{embedding}\right) \tag{4}$$
$$where\ W_{embedding}\ \&\ b_{embedding}\ are\ weights\ and\ biases$$

The values of I_n change during the training of the network, and finally we get a unique d dimensional representation for each word.

3. Convolutional neural network

The advantage of convolutional neural networks is that they use parts of inputs and perform convolutions instead of using the entire input at once. [6] showed that CNN's can also be used with sentences by using sliding window approach. We use the same sliding window approach of fixed window size k on a single layered CNN. Given the set of indices of a record along with padding, the window slides through the input from beginning to end performing convolutions at each position. Each convolution produces estimations regarding which label the words in the window belong. Once the estimations for each position of window are obtained, they are averaged in the softmax layer. Finally the label which gets the highest probability is most likely appropriate label for the input. The following figure is the complete model of the architecture:

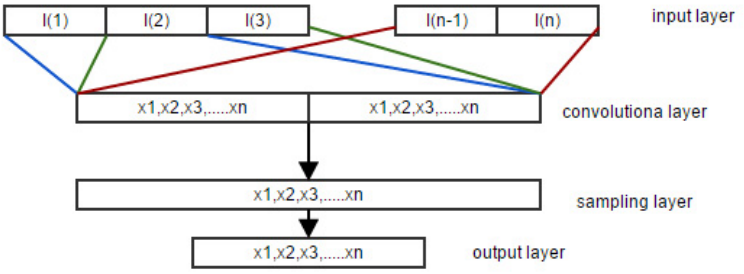

Fig 1: Architecture for machine part of hybrid machine-human approach

The convolutional layer outputs for each position of window is given by:

$$sigmoid(\sum W_c^T x + b_c) \tag{5}$$

The sampling layer output is given by:

$$softmax(\sum W_s^T x + b_s) \tag{6}$$

The initial values for $W_{embedding}, W_c, W_s, b_{embedding}, b_c, b_s$ are chosen randomly in the range $(0,1)$. The training uses back propagation algorithm to update the weights and biases. The final weights and biases defines a complex function which identifies various combinations of the words to their labels.

4. Crowdsourcing

Crowdsourcing, as discussed in section 1, involves HIT's which will be solved by the crowd. It comes under the human part of the hybrid machine-human model. The output of the CNN contains records with predicted labels. Crowd enabled query processing systems such as Qurk [7] and crowdDB [8] use simple queries to help perform entity resolution.

SELECT * FROM products p, product q WHERE p.product_label = q.product_label

The above query implies that all the records under a label are identical records. When considering huge datasets, there will be certain range of error even for CNN. We obviously do not have to consider the records which got a strong probability for a label. However, the records for which the probabilities of labels are only minor differences there is a high chance that the predicted label has error. To solve these uncertainties, we chose a certain threshold for probabilities and created record pairs. These record pairs are uploaded on the crowd platform and sent to the crowd. Approaches for creating HIT's are discussed in [5] and for this paper, we only have to note that the more accurate the output of machine part is, the lesser the number of HIT's that will be given to the crowd.

5. Experimental results

We used dataset provided by the Boeing which includes sensor descriptions. The goal is to map them to ATA labels to improve airplane health management. The ATA labels are the standard references used to describe various sections of the plane. For example ATA 28 has all information related to fuel. Obviously, the sensor descriptions, even though they belong to same ATA label may not have much similarities. In such a case we cannot use simple similarity functions to identify which records are identical. Therefore, we definitely need a trained network to identify which combination of words in the descriptions can correctly identify the ATA label.

Since the descriptions had more technical terms, we created keywords by removing the common English words that did not represent the records uniquely. We then created indices as described in section 3.2. For the training of CNN, we used a window size of 2. Also experimentally, we observed that word embedding of dimension 50 gave good results. The following are the values of various parameters used:

$$number\ of\ records = 1950$$
$$I = 1346 \rightarrow number\ of\ keywords\ in\ the\ dataset$$
$$d = 50 \rightarrow dimension\ of\ each\ word\ vector$$
$$k = 2 \rightarrow sliding\ window\ dimension$$

We used 1000 records for training, 650 records for validation and remaining 300 records for testing. In order to compare the results with Jaccard similarity, we used the same 300 records for Jaccard similarity.

The following table shows the comparison between Jaccard similarity and CNN

Table 2. Experimental Results

parameter	Jaccard similarity	CNN
Number of records classified correctly	123	166
Number of records classified incorrectly	177	134
Time taken in mins	1.27	8.6

If we use Jaccard similarity we need to perform 210925 record pair comparisons. Using this approach we observed that 223 record pairs crossed the threshold of 0.6. Among these 46 of them are considered similar even though they do not belong to the same label. Considering the percentage of misclassifications, even if these records are uploaded as tasks on crowd platform it won't be efficient as the HIT's have to be generated based on the number of correct classifications.

By using CNN, 43% of records are considered as not similar and 53.3% of them are classified correctly. This result is much better that Jaccard similarity. Moreover, if use the 0.7 as uncertainty threshold i.e. records having probabilities for labels more than 0.7 are guaranteed to be similar then the number of HIT's to be created will less than those that have to be generated using Jaccard similarity.

6. Conclusion and future work

In this paper, we showed how entity resolution can be performed using hybrid machine-human approach on a labeled dataset whose duplicate records have low similarity. By using CNN in place of similarity function, we proved that it not only reduces the number of HIT's but also reduces the number of misclassifications. This application of CNN for entity resolution showed that deep learning algorithms can be used for finding duplicate records in a dataset. Since we use only one convolutional layer, the efficiency reached only 53.3%. Hence, the future work will include the training of CNN with multiple convolutional layers to improve the efficiency. In addition to that, we also plan to build a dynamic crowdsourcing platform which also helps to reduce the tasks given to the crowd.

7. References

[1]H. B. Newcombe, Handbook of record linkage: methods for health and statistical studies, administration, and business: Oxford University Press, Inc., 1988.

[2] H. Newcombe, J. Kennedy, S. Axford, and A. James, "Automatic linkage of vital records," in Record linkage techniques, 1985: proceedings of the Workshop on Exact Matching Methodologies, Arlington, Virginia, May 9-10, 1985: co-sponsored with the Washington Statistical Society and the Federal Committee on Statistical Methodology, 1986, p. 7.

[3] I. P. Fellegi and A. B. Sunter, "A theory for record linkage," Journal of the American Statistical Association, vol. 64, pp. 1183-1210, 1969.

[4] D. R. Wilson, "Beyond probabilistic record linkage: Using neural networks and complex features to improve genealogical record linkage," in Neural Networks (IJCNN), The 2011 International Joint Conference on, 2011, pp. 9-14.

[5] J. Wang, T. Kraska, M. J. Franklin, and J. Feng, "Crowder: Crowdsourcing entity resolution," Proceedings of the VLDB Endowment, vol. 5, pp. 1483-1494, 2012.

[6] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in Proceedings of the 25th international conference on Machine learning, 2008, pp. 160-167.

[7] A. Marcus, E. Wu, D. R. Karger, S. Madden, and R. C. Miller, "Crowdsourced databases: Query processing with people," 2011.

[8] M. J. Franklin, D. Kossmann, T. Kraska, S. Ramesh, and R. Xin, "CrowdDB: answering queries with crowdsourcing," in Proceedings of the 2011 ACM SIGMOD International Conference on Management of data, 2011, pp. 61-72.