Title: Deep Learning on Entity Matching using Structured Data

Mustafa Barez

Abstract

Entity matching (EM) is a process that determines whether data instances refer to the same real-world entity, and recent studies have suggested deep learning (DL) plays a significant role in this field. The following paper displays different methods in which DL algorithms are manipulated through structured data. Experiments were performed using the DeepMatcher algorithm and its datasets and were taken from the Github repository. The experiments assessed the algorithm specifically for accuracy through F_1 scores. Two of the datasets were then taken and reduced in size to minimize overfitting and to determine whether or not reductions in size or the number of attributes would produce significant performances in accuracy. The results saw minimal improvements when the data size was condensed and fewer attributes were available. This paper contributes to the study of DL algorithms by providing a detailed review of how they assist in EM problems and how they are affected by manipulations in structured data.

Keywords*: Entity matching, deep learning, DeepMatcher, structured data*

Introduction

One way of integrating multiple sources of data is through Entity Matching (EM) [1]. EM decides if two given mentions in the data refer to the same real-world entity [3] and play a role areas such as data cleaning and integration. In recent years, EM has seen a rise in other areas that can assist in the data discovery. One is through a section of machine learning known as deep learning (DL). DL utilizes multiple layers to produce features from given inputs and use neural networks. These neural networks are used to cluster and classify data and can also extract features given to other algorithms. This plays a role in problems such as filtering, fraud detection, and customer relationship management [5].

There are three distinct types of EM problems that DL can play a role in solving. Structured data, this is where two datasets have the same schema with attributes correctly aligned and data containing numbers or text with slight variations from one another [6]. Textual data, these are problems that consist of text entries such as the description of a person. Dirty data, these are problems with datasets of the same schema but possess attribute values that may be under the incorrect attribute [8]. Figure 1 displays a detailed example of structured data as the present study focuses specifically on structured data for EM and uses the hybrid DL model. Previous studies have shown that structured data performs comparably with Magellan and hybrid models provide the best performance compared to the other three DL models [4].

Previous research has proposed the question of whether or not DL plays a role in EM. One of the areas that DL can assist in EM problems is through structured data. DL would ignore the attribute boundaries thus reducing them to matching instances.

Student Name	Number	Age
John Smith	134	19

Student Name	Number	Age
J. Smith	134	19

Structured Data

Figure 1. Examples of Structured data used for EM.

This research uses the Magellan system as it supports EM and contains a set of packages in Python's data science stream. It has been proven to be apply successfully to EM projects and is open source. The code in this paper was taken from the Magellan website [2] and is called DeepMatcher. This is a python package used to perform EM through DL methods and uses pre-built neural networks to train and apply DL models [7]. These DL models are then used for EM and consists of four critical steps: data processing, specifying the neural network, training the model, and applying it.

The present study seeks to understand the accuracy of the DeepMatcher algorithms in relation to the data evaluated. The contributions of this paper are as follows:

- Tested five structured datasets on the DeepMatcher algorithms and provided a comprehensive review of how these algorithms are influenced by changes in this data. Following this, two of the five datasets were used to test the algorithms once more to observe the response of the algorithms.

- A study displaying how DL algorithms, specifically those of the hybrid model, react to condensed versions of data to reduce overfitting. It was determined that condensed versions of similar datasets provide little variation in terms of accuracy of the DL models.

- An analysis of the DeepMatcher algorithms and their performance in terms of accuracy through F_1 scores as well as discussions for future areas of research.

Experiments

A number of experiments were performed on particular datasets in order to evaluate how the DeepMatcher algorithm reacts to different types of structured data and the accuracy levels presented. After conducting accuracy results, the percentages were compared for analysis to provide a clear sense of which data DL methods provide quality results for (those with percentages of 75% or higher). Pairs of tuples will be matched to determine whether they refer to the same real-world entity. Labeled examples will be used and taken as input and will be trained in the neural network using supervised learning. After the neural networks are trained, unlabelled tuples will be applied to create predictions.

DeepMatcher works in a two-step process to perform an EM task given two tables. The first step is blocking on two tables as this removes tuple-pairs that are far different from one another to get a candidate set which is then used to perform the next step, matching. This step predicts each pair in the candidate set. The next step consists of four sub-steps where Magellan takes a sample from the candidate set, and the user labels all the pairs from the sample set as the training data. A classifier then learns based on the sample, and the user applies to classifier to the candidate set to predict pairs as a match or non-match [9].

The following experiments evaluated the hybrid DL model and were implemented through Torch, a DL framework that can be used for accelerated training and which utilizes GPUs [10]. The source code was taken from the DeepMatcher repository on the Github website and ran on a HP Notebook x64-based processor with 12.0 GB RAM. The algorithms were run on the jupyter notebook to provide a platform form analysis.

Results

Dataset	Size	Number of Attributes	F_1 Score for Hybrid Model
iTunes – Amazon	539	8	89.2
BeerAdvo – RateBeer	450	4	75.0
Fodors – Zagats	946	6	99.6
Walmart – Amazon	10,242	5	68.4
Amazon – Google	11,460	3	71.1

Table 1. The following datasets were taken from the DeepMatcher repository. The DeepMatcher python algorithms were run on the datasets and the F_1 values were recorded along with the size, domain, and number of attributes of the dataset.

Dataset	Size	Number of Attributes	F_1 Score for Hybrid Model	Improvements in terms of percentage
Walmart – Amazon	8,242	3	73.6	7.60
Amazon – Google	9,460	3	72.8	2.40

Table 2. A simplified dataset for determining whether or not the new data contributes to a minimization in overfitting. The new Walmart – Amazon data set had two fewer attributes and the size reduced by 2,000. The Amazon – Google dataset had the same number of attributes with the size also reduced by 2,000.

When running the algorithms on the datasets, the F_1 scores were noted from the jupyter notebook. The F_1 scores were the return value of the run_train method used to train the model for the specific dataset. The value represents the measure of the test's accuracy and uses precision and recall in the formula $F_1 = 2 * \frac{precision * recall}{precision + recall}$.

Discussion

The experiments revealed how structured data affects the DeepMatcher algorithms that use the hybrid model. The Walmart – Amazon dataset provided the lowest F_1 score due to the size of the dataset as well as the overfitting presented in the model. Overfitting is inferred based on the quality of the training data used for the model which provides some insights into how these algorithms can be manipulated. With the range in the F_1 scores going from 68.4 to 99.6, the sensitivity of these DL algorithms are noted based on the size of the

dataset but are not correlated based on the number of attributes.

The first three datasets in Table 1 display results with relatively high F_1 scores in comparison to the last two datasets. This is due to the quality of the data presented as well as the size. The question of size causes variations as the third dataset, Fodors – Zagat is almost twice as much as the first two and presents the highest F_1 score. The reason for discrepancy in these scores would point towards the number of attributes of the first dataset as well as the quality of data in all three datasets. The data quality of the Fodor – Zagat dataset was relatively clean prior to running the algorithms. It contained few errors, allowing the EM methods from DeepMatcher to run efficiently.

The Amazon – Google dataset had an F_1 score that was also affected by the size of the dataset and is why it presents the second-lowest score of 71.1 according to Table 1. The dataset also presented signs of overfitting in the model as the training set provided better results than the test set. In order to get a sense of whether a reduction in overfitting can play a significant role in increasing the accuracy of predictions, the Walmart – Amazon and Amazon – Google datasets were simplified through decreases in overall size and number of attributes. The simplified Walmart – Amazon dataset displayed higher improvements in accuracy which was expected due to the significant changes in the number of attributes as well as a reduction in the amount of data. The other simplified dataset, Amazon – Google displayed an improvement, however, the percentage-wise was significantly smaller. This was due to the fact that the dataset still required cleaning in order to achieve high accuracy because DeepMatcher's neural networks could not classify the labels accordingly. The overall results of this display the lack of sensitivity to reductions in data in terms of how they affect the DeepMatcher algorithms. Future research must be conducted on other algorithms to determine whether or not this trend is common in situations that require DL processes.

Future Work

Several paths can be taken from the present study. One of the first areas would be experiments concerning other models of neural networks since the present focused solely on the hybrid one. By focusing the research on a specific type of DL method, results provide more accurate conclusions as to how data affects the given type. Another area for research would be to focus on the type of training data available. The present study focused on structured data since previous research found that DL solutions with this type of data provided significant and competitive results with EM tasks. Future research may focus on other areas including dirty, or textual data and compare the roles of DL on these types of datasets with structured data. A separate route involves adding machine learning algorithms such as bagging to the datasets displayed signs of overfitting. By including these techniques to reduce overfitting, one can better analyze how test data affects the DL algorithms. Creating multiple ways for manipulation would provide further insights as to how these algorithms work with test data.

Conclusion

The study examined how DL affects EM tasks and focused specifically on structured data using the hybrid DL model. The DeepMatcher algorithms and five datasets were taken from the Github repository and assessed to display how DL algorithms are affected by changes in structured test data. After initial results, two of the datasets which presented signs of overfitting had their data simplified in order to determine whether or not this played a significant role in the model's accuracy. It was found that the results of the simplified datasets presented more accurate scores, however, the significance is minimal. Future studies may utilize the given datasets with complex machine learning algorithms, such as bagging, in order to gain a better understanding of the strengths, weaknesses, and areas for manipulation in DL algorithms.

References

[1] Shen, W., DeRose, P., Vu, L., Doan, A., & Ramakrishnan, R. (2007, April). Source-aware entity matching: a compositional approach. In *2007 IEEE 23rd International Conference on Data Engineering* (pp. 196-205). IEEE.

[2] Konda, P., Das, S., Suganthan GC, P., Doan, A., Ardalan, A., Ballard, J. R., ... & Prasad, S. (2016). Magellan: Toward building entity matching management systems. *Proceedings of the VLDB Endowment*, *9*(12), 1197-1208.

[3] Shen, W., Li, X., & Doan, A. (2005, July). Constraint-based entity matching. In *AAAI* (pp. 862-867).

[4] Sidharth Mudgal et al. 2018. Deep Learning For Entity Matching: A Design Space Exploration. Technical Report. http://pages.cs.wisc.edu/~anhai/papers/ deepmatcher-tr.pdf.

[5] Chris Nicholson. 2019. A Beginner's Guide to Neural Networks and Deep Learning. (2019). Retrieved January 4, 2020 from https://pathmind.com/wiki/neural-network

[6] Arvin Arasu. Extracting Structured Data from Web Pages. Retrieved January 3, 2020 from http://infolab.stanford.edu/~arvind/papers/extract-sigmod03.pdf

[7] Revaud, J., Weinzaepfel, P., Harchaoui, Z., & Schmid, C. (2016). Deepmatching: Hierarchical deformable dense matching. *International Journal of Computer Vision*, *120*(3), 300-323.

[8] Kim, W., Choi, B. J., Hong, E. K., Kim, S. K., & Lee, D. (2003). A taxonomy of dirty data. *Data mining and knowledge discovery*, *7*(1), 81-99.

[9] Nie, H., Han, X., He, B., Sun, L., Chen, B., Zhang, W., ... & Kong, H. (2019, November). Deep Sequence-to-Sequence Entity Matching for Heterogeneous Entity Resolution. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (pp. 629-638). ACM.

[10] Kovalev, V., Kalinovsky, A., & Kovalev, S. (2016). Deep learning with theano, torch, caffe, tensorflow, and deeplearning4j: Which one is the best in speed and accuracy?.