Deep Learning vs Multidimensional Classification in Human-Guided Text Mining

MARAT ZHANIKEEV[1,a)]

Abstract: Deep (Neural) Learning has recently become popular in AI research. The method is traditionally showcased in vision-related tasks where input can be easily regulated. However, when applied to text mining, the irregular textual input becomes a hurdle. Overcoming the hurdle involves processing the text and using its frequency distribution as a numeric input. This paper compares the technology with a recently proposed method in multidimensional classification. The specific feature in focus is a human-guided system where the learning dataset is not available at once but arrives gradually, along with human annotation.

Keywords: deep learning, multidimensional classification, human-guided learning, metromap classifier, classifier chains, text mining

1. Introduction

Classification today is a very rich topic with many individual methods as well as classes of methods [4]. However, the *semi-supervised learning* class of methods is arguably the most popular today. Both the *multidimensional classification* (MDC) and *Deep Learning* (DL) methods belong to this class of methods. This is convenient because is means that this paper does not need to present a full taxonomy of the various learning methods.

The two specific methods discussed and compared in this paper are the **Metromap Classifier** (MC) which belongs to the group of multidimensional classifiers [6] along with Classifier Chains (CC) [7] and the general graphical approach to classification [5], and the general *Deep Learning* (DL) method. The latter is a very active area of current research [10]. There is some branching into research on the internal structure of the underlying neural networks [9]. However, it is important to remember that DL has been shown to be unreliable on rare yet key occasions [8]. In a manner of speaking, this paper can be viewed as a verification of DL in the context of *text mining*. Note that this context is not common in DL research which mostly focuses on vision.

The Metromap Classifier (MC) – also referred to as the Metromap Method in this paper [2] – proposes yet a different kind of supervision in learning, the one that is based on continuous human feedback in learning. The context also assumes that learning is performed on top of a *folksonomy*, in which, by definition, tags and the context in general are fuzzy. The context also assumes that a classifier is trained to find *blackswans* [3], which are defined as rare but very high-impact events. Note that blackswans are in fact the context in which DL has been found unreliable in recent literature [8]. The final component of the context in this paper is the use of a *visual structure* for communication between humans and classifiers – the latter referred to as *robots* in this paper. The actual visual structure used by teh MC is the metromap [1] – hence the name of the method. With metromaps, the classifier can be viewed as a kind of Classifier Chain (CC) but with a more socially clear meaning attributed to the combinations of individual Binary Relevance (BR) classifiers in a combination [2]. Metromaps are used in both directions, providing both the context for the classifier and serving as human feedback.

The main objective of this paper is to compare MC with DL in the context of text mining. The MC in this paper is represented by the recent results in [2] which show 70-80% success rates compared to the ideal case of relying only on human operators. Note that text mining is not a common application of DL and requires additional effort to make DL applicable to text. Specifically, the text is pre-processed and converted into a numeric form suitable as input to a neural network. Even more importantly, additional effort is spent to make sure that input is in regular size across all data tuples – another strict condition to make DL applicable in practice.

2. The Metromap Process

More details on the Metromap Classifier (MC) can be found in [2]. This section provides only the basic outline of the method as far as is necessary for the later comparison with the DL method.

The basics of the MC method is as follows. The metromap is used as the core visual structure for the entire method [1]. Although the structure and its use are different, existing methods in MDC are known to use visual structures [4][5]. The method depends on the use of simple Binary Relevance (BR) classifiers in a manner similar to that used by the Classifier Chain (CC) method [7]. However, the similarity ends there as the MC method relies on an entirely different use of the metromaps. Specifically, the

[1] Computer Science and Systems Engineering
Kyushu Institute of Technology
Kawazu 680-4, Iizuka-shi, Fukuoka-ken, 820–8502 Japan
[a)] maratishe@gmail.com

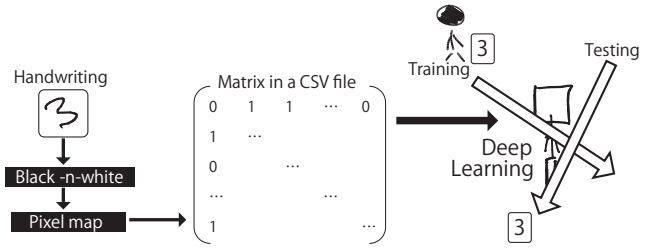

Fig. 1 The traditional Deep Learning process, commonly applied to matrices representing image bitmaps.

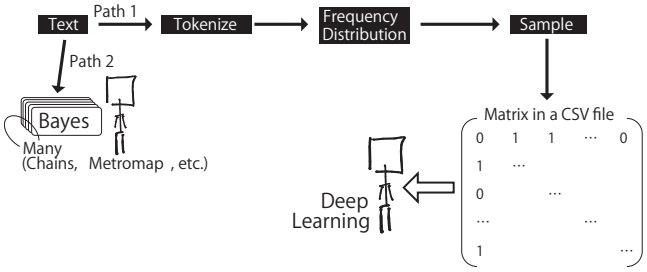

Fig. 2 Both metromap (and generally multidimensional classification) and Deep Learning processes in the context of text mining.

MC method does not depend on the order of BR classes, instead applying a selection algorithm based on the connections identified from the metromap. The algorithm strives to counteract the fuzziness of outcome from BR classes and maximize the number of hits.

The results for the MC method show very good performance. In fact, it was found better than the existing MDC methods, with only about 20% of wrong outcomes compared to the human operator. These results are discussed further in this paper as part of the comparison with the DL results.

3. The Deep Learning Process

This section provides the background on the traditional Deep Learning (DL) process. As was mentioned earlier, DL methods are commonly expected to operate in image recognition (vision) settings and are not commonly applied to textual input [8]. For example, DL methods are commonly tested on the MNIST dataset of handwritten digits [12].

Fig.1 shows the DL process divided into two parts: in the first part graphics are converted to a numeric matrix, which is then fed to DL neural networks for analysis.

DL poses a problem for any input, not just the text as is known in this paper. Even graphical input has to be converted, although the method for graphics is much more straightforward. Note that MDC methods are under to such restriction and can be therefore considered as more flexible.

The left side of Fig.1 shows the standard conversion process. The image first is converted to a black-and-white image. Then, having scaled the image to an appropriate size, individual pixels are converted to individual numbers. It is possible to use single digits for that but it is much more common to use only 0 and 1 for each pixel. The result of this pre-processing is a matrix as is shown in Fig.1. Note that the matrix notation might be confusing by hinting that the rows in the matrix become separate data tuples in DL input. In reality, the entire contents of them matrix is treated as **one tuple** when fed to a neural network. The order of rows and columns is not important as long as the consistent order is used across different tuples and across training/validation cycles.

The right side of Fig.1 shows the two main uses of the resulting matrix. In *training*, human provides metadata for the contents of the matrix (number 3 in the figure), which DL uses for learning and incorporates its into its current model. In *testing*, DL uses its current model to predict the metadata (number) that a given graphical data represents.

As the figure shows, DL input commonly comes in form of a CSV file. It is expected that the metadata is included as the last column in the file. DL uses it as feedback during training and for validation in testing.

4. The Text Mining Context

Repeating the earlier statement, a major difference between DL and MC methods is that DL is mostly applied to images while MC (and largely MDC) can be applied to any input, including text. This paper specifically compares the two methods in the context of text mining.

As was done to images in the previous section, text also should be pre-processes become it can be used as input to a neural network. This pre-processing is trickier because the text is expected to be converted into numbers. Even more trickier is the part where DL requires that all input is of the same size.

Fig.2 shows the 2 ways to classify text, based on DL (left-to-right) and MC (down).

The following basic process is required for DL.

Step 1: Tokenize. This is a trivial step but requries different algorithms depending on language and possibly the nature of textual input. At the end of this step, we get a list of tokens, possibly of variable length (English).

Step 2: Word Count and Distribution. All tokens are then counted and counts used to create a frequency distribution for tokens in the entire text. It is common to list the values in decreasing order. At this stage, the textual information is converted into numbers but the size of the list can vary depending on text.

Step 3: Sample Distribution. This part is necessary to provide input of a regular size. For this, the frequency distribution is sampled at a fixed step (this paper uses 1%), resulting in a fixed-length list of counts for tokens with frequency of 1%, 2%, and so on. The new distribution is more practical when generated in a cumulative way, i.e. at least 1%, 2%, etc.

At the end of this process, we have 100 values (at 1% sampling step). Since DL requires a single digit input, we can quantize the sampled frequency to map it to a scale between 1 and 9. The final list becomes a matrix (same as for images, for convenience) but is handled as a flat list of values when used as input to a neural network.

The MC process is the same as is described in [2] Specifically, multiple BR classifiers are used independently and a metromap-based logic is applied to produce a single-digit outcome.

One major note is due on tokenization and frequency distribution. Naive Bayes Classifiers also tokenize input and analyze occurrence frequency. However, the parallel with DL methods is

very weak because the core methodology in the MC method is found in the post-processing of individual BR outputs. By contrast, the above pre-processing is the signal proper for the DL methods and is therefore part of the core methodology.

5. Analysis Setup

This paper uses th e same dataset as in [2], which is paper metadata from a major scientific portal. The multiple dimensions are *topics*, of which one or more can be used by the human operator to assign to each paper. The two special dimensions are *hot* and *cold*, which are also used as flags in testing. The two important cases are *cold=yes,hot=yes*, which means that human operator would like to look at the paper but use it as part of its own research, and *cold=no,hot=yes* which represents a blackswan – the rare case of a paper which the human operator actually needs to find/read and use it afterwards.

The metadata is used in three distinct combinations: *title* only, *title and keywords*, and *title and keywords and abstract*. Each combination is analyzed separately to visualize the difference in classification results depending on the volume and type of metadata.

The following process is used for both MC and DL methods. Human makes the initial assignment of all tags, including *cold* and *hot* flags described above. For the MC method, the robot starts making predictions from the 2nd round, while DL predictions start after 10 idle rounds – this setup is arbitrary and does not affect the performance in the long run. In each next round, both human and robot perform the assignment and the robot uses human feedback to compare with its own prediction as well as use human feedback to update its current model.

Note that ideally the final objective of such a system is full automation. Specifically, robots are built and trained for finding *blackswans* [3]. In context of papers, blackswans are the very few papers a given reader would use as part of his/her own literature (*cold=no,hot=yes*).

Results are visualized as curves, one for human/ideal outcomes and the other for the hits scored by one of the two compared classifiers.

6. Analysis of the Metromap Process

Fig.3 shows the results from the MC tests [2]. The performance is interesting from the viewpoint of comparing them to the DL performance in the next section. Each plot in the future is for a randomly selected round for a distinct combination of metadata. See [2] for the definition of Dumb Classifier used for comparison.

The top plot is for *title only* metadata. Both the classifiers are good, with very little difference between the two. The middle plot shows a major gap between the two methods, with the MC method leading by 20-30%. This performance can be explained by the fact that the use of keywords adds too much noise to the already fuzzy titles. The same situation is found for the bottom plot, where adding the abstract to metadata does not affect the performance to any noticeable degree.

One main feature for all the plots is that the MC method shows the constant performance of about 20-30% below the human (ideal) assignment. This performance is stable and does not depend on the metadata mix for the MC method, while performance of the traditional method can vary wildly.

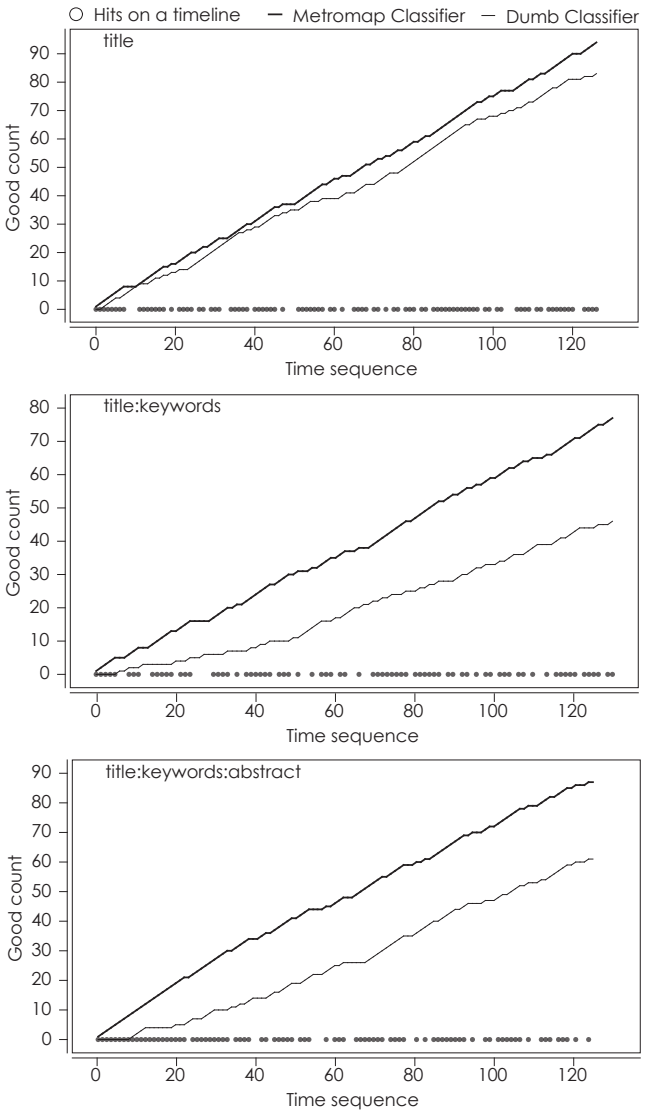

Fig. 3 Results for the metromap process represented as selected learning curves.

Note that performance for both methods is linear with time. This does not directly relate to this paper, but it is an indication that the *blackswan strategy* has failed because the curve should be a concave in order to be able to find blackswans automatically after a substantial period of learning. This topic will be researched in depth in future publications.

7. Analysis of Deep Learning

Fig.4 shows several selected cases for the DL method. The set of conditions is richer than the one in the previous section because this time both *metadata mix* and *flags* are used for selection. The plots show only a subset of all the possible combinations of the two. The view this time is also more regular, exactly 100 runs for each plot. It should be reminded that DL starts making predictions from 11th round.

The 1st (topmost) plot is for title only and the broader flag coverage (*cold=yes,hot=yes*). DL shows the worst performance for this method, with only 10 hits out of 100. 2nd and 3rd plots show

better performance, at roughly 35-40% of scored hits. Note that the metadata mix is different for these plots, which means that adding the abstract does not change the performance to any noticeable degree – the same was found earlier for the MC method. However, the 1st plot used *title only* which might be another fact affecting the performance.

Interestingly, the 4th plot uses roughly the same conditions as 2nd and 3rd, but shows about 10% lower performance. This means that performance of DL classifications depends on a specific subset of documents. In other words, these plots indicate that DL is highly volatile and can produce unexpected results.

As an overall evaluation, DL performance is roughly at the level of 10-40%, depending on conditions and parameters. This is very low because previous chapter showed 70-80% hit scores, which is double the DL rate.

8. Conclusion

This paper is the first known attempt to compare Deep Learning with Multidimensional Classification in context of text mining. In current practice, the two methods rarely clash because Deep Learning is commonly applied to image recognition while Multidimensional Classifiers are normally used for text.

This paper shows that text is not a natural input for Deep Learning and proposes a simple pre-processing algorithm for textual input based on frequency distribution sampled into a set of a regular size suitable for input into a neural network.

Results show that while Multidimensional Classifiers show 70-80% hit scores, the standard Deep Learning engine achieved only half that rate. Future work will attempt to inverse the problem and compare the two methods in context of image recognition.

References

[1] M.Zhanikeev, "On Context Management Using Metro Maps", 7th IEEE International Conference on Service Oriented Computing and Applications (SOCA), Matsue, Japan, November 2014.
[2] M.Zhanikeev, "Multidimensional Classification Automation with Human Interface based on Metromaps", 4th International Congress on Advanced Applied Informatics, Okayama, Japan, July 2015.
[3] M.Zhanikeev, "Black Swan Disaster Scenarios", IEICE Technical Report on Pattern Recognition and Media Understanding (PRMU), vol.114, no.356, pp.45–48, December 2014.
[4] X.Zhu, A.Goldberg, Introduction to Semi-Supervised Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan and Claypool Publishers, 2009.
[5] D.Koller, N.Friedman, Probabilistic Graphical Models: Principles and Techniques. Adaptive Computation and Machine Learning, MIT Press, 2009.
[6] J.Ortigosa-Hernandez, J.Rodriguez, L.Alzate, I.Inza, J.Lozano, "A Semi-supervised Approach to Multi-dimensional Classification with Application to Sentiment Analysis", 6th Spanish Workshop on Data Mining and Learning (TAMIDA), pp.129–138, 2010.
[7] J.Read, B.Pfahringer, G.Holmes, E.Frank, "Classifier chains for multi-label classification", Machine Learning, Springer, vol.85, issue 3, pp.333–359, June 2011.
[8] A.Nguyen , J.Yosinski, J.Clune, "Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images", IEEE Computer Vision and Pattern Recognition (CVPR), 2015.
[9] J.Weston, F.Ratle, H.Mobahi, R.Collobert, "Deep Learning via Semi-Supervised Embedding", Neural Networks: Trick of the Trade, Springer LNCS vol.7700, pp.639–655, 2012.
[10] G.Goos, J.Hartmanis, J.Leeuwen, Neural Networks: Tricks of the Trade. Springer LNCS vol.7700, 2nd edition, 2012.
[11] H2O: R Package for Learning Algorithms. [Online]. Available: http://cran.r-project.org/web/packages/h2o (June 2015)
[12] MNIST Dataset of Handwritten Digits. [Online]. Available: http://yann.lecun.com/exdb/mnist/ (June 2015)

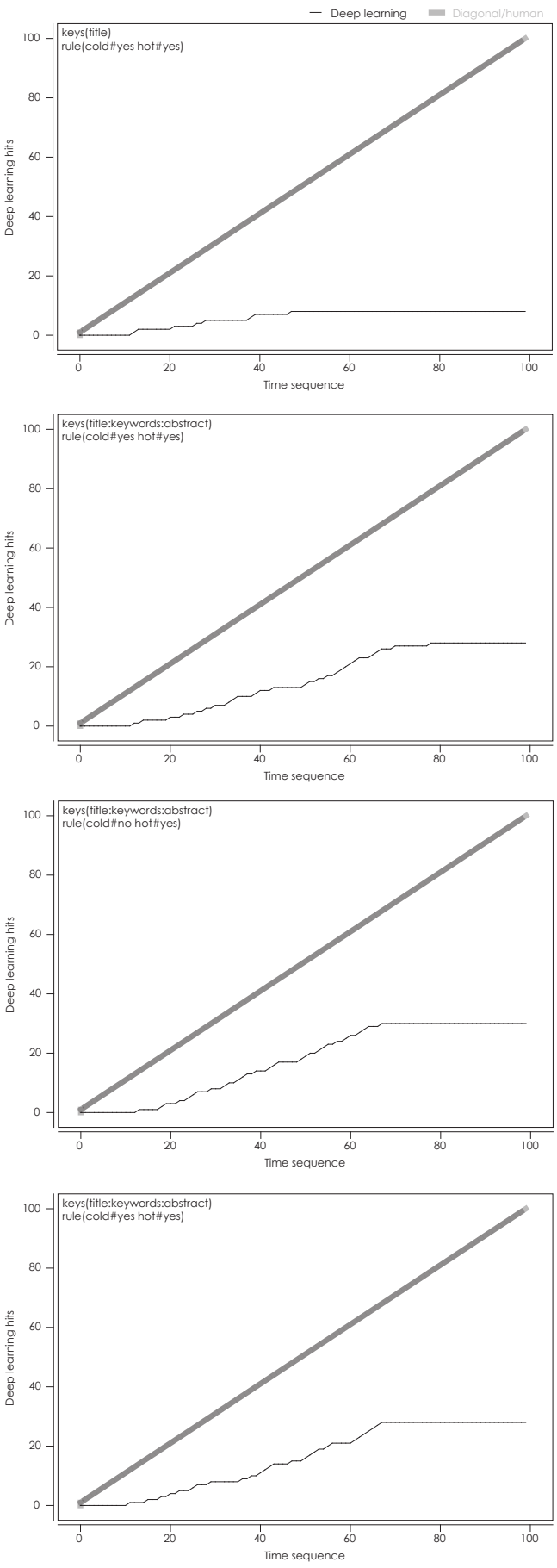

Fig. 4 Results for the Deep Learning process represented by selected learning curves.