Health Informatics Meets eHealth
D. Hayn and G. Schreier (Eds.)

doi:10.3233/978-1-61499-759-7-8

Word Sense Disambiguation of Medical Terms via Recurrent Convolutional Neural Networks

Sven FESTAG[a,1] and Cord SPRECKELSEN[a]
[a]*Department of Medical Informatics, Medical Faculty, RWTH Aachen University*

Abstract. Background: Tagging text data with codes representing biomedical concepts plays an important role in medical data management and analysis. A problem occurs if there are ambiguous words linked to several concepts. Objectives and Methods: This study aims at investigating word sense disambiguation based on word embedding and recurrent convolutional neural networks. The study focuses on terms mapped to multiple concepts of the Unified Medical Language System (UMLS). Results: We created 20 text processing pipelines trained on a subset of the MeSH Word Sense Disambiguation (MSH WSD) data set, each pipeline disambiguating the sense of one word. The pipelines were then tested on a disjoint subset of MSH WSD data. Most pipelines achieved good or even excellent results (70% of the pipelines achieved at least 90% accuracy, 40% achieved at least 98% accuracy). One poor-performing outlier was detected. Conclusion: The proposed approach can serve as a basis for an up-scaled system combining pipelines for many ambiguous words. The methods used here recently proved very successful in other fields of text understanding and can be expected to scale-up with improved availability of training data.

Keywords. Neural Networks (Computer), Unified Medical Language System

1. Introduction

Coding of medical terms from full text data is a crucial prerequisite for medical data management in clinical research and in patient care as well. Scientific publications, for instance, are indexed by controlled keywords (e.g. the Medical Subject Headings used by PubMed). Moreover, full text entries of patient records are processed and further analyzed applying automatic entity recognition. In all these cases medical terms or phrases used in a given text are assigned codes/controlled vocabulary representing the biomedical concept behind these expressions. An obvious and well-known problem is that biomedical texts often contain ambiguous abbreviations and words. I.e. single expressions denote different concepts (homonymity) and, therefore, need to be assigned different codes depending on the context. A common example is the term *cold*, which may be an adjective describing the temperature, a noun referring to the common cold, or the acronym for the Chronic Obstructive Lung Disease (COLD).

Word sense disambiguation (WSD) aims at inferring the correct meaning of a given term depending on the surrounding text [1]. WSD (re-)establishes a functional mapping between terms (plus surrounding text) and concepts.

[1] Corresponding Author: Sven Festag, Department of Medical Informatics, Medical Faculty, RWTH Aachen University, Pauwelsstr. 30, 52074 Aachen, Germany, E-Mail: sven.festag@rwth-aachen.de

The Unified Medical Language System (UMLS) Metathesaurus was implemented to enable a terminological cross-walk between different medical terminology systems. Entries of the different vocabularies are assigned UMLS Concept Identifiers (CUI) representing the unique concept. The UMLS also captures the ambiguities of medical language by assigning more than one CUI to a given term. The level 0 subset of the UMLS 2016 version contains 78,918 different, ambiguous terms in its ambiguity index (AMBIGLUI).

While the UMLS was adopted to coding tasks soon after its first implementation a seminal approach by Rindflesch and Aronson considered disambiguation concerning UMLS Metathesaurus concepts [2]. Their rule-based system heavily depends on the UMLS semantic network.

The authors of [3] and [4] describe the use of statistical learning approaches for the same purpose. However, the focus of both studies lay on the automatic creation of a corpus for training and evaluation of machine learning methods for WSD. A similar goal was pursued by Stevenson et al. [5]. This focus also shows the importance of labeled training data for supervised learning systems.

Knowledge-based or graph-based methods like Journal Descriptor Indexing and Machine Readable Dictionary, neither of both depending on supervised training, are addressed in [6,7].

In 2011 a benchmark test set, the MeSH Word Sense Disambiguation (MSH WSD) data set, was published by Jimeno-Yepes et al. for the evaluation of WSD systems in the biomedical domai [8]. An evaluation of a Naïve Bayes approach for WSD using this data set was conducted by Plaza et al. [6].

Our investigations address the use of more recent supervised machine learning approaches for WSD in biomedicine. Most recently deep learning techniques, namely convolutional neural networks, have achieved striking success in several areas of natural language processing, such as machine translation [9] or speaker-independent speech recognition [10]. Word vector embedding and (recurrent) convolutional neural network, have proved very effective in the task of text classification in general [11]. Thus, systematic investigation of similar approaches in the field of medical WSD seemed relevant.

1.1. Aim of The Study

This study aimed at evaluating the use of word embedding, hereinafter also referred to as vector representation of words, and recurrent convolutional neural networks for WSD in the biomedical domain. More precisely, we investigated the accuracy achieved by that approach on the task of assigning the correct UMLS Metathesaurus concept (CUI) to an ambiguous word given its context, i.e. the text surrounding the word. The evaluation was based on a subset (strictly separated from the training data subset) of the MSH WSD data mentioned above.

2. Methods

The proposed method adopts vector representation of words introduced by Quoc and Mikolov [12] and recurrent convolutional neural networks (RCNN) for text classification

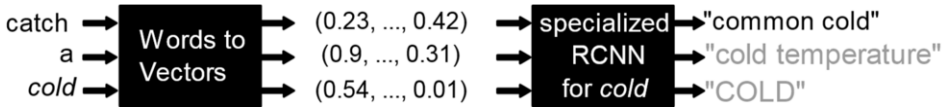

Figure 1. Example pipeline for the WSD of "cold"

presented by Lai et al. [13]. Both methods factor the context of words into the computation and, thus, keep the most important information needed for disambiguation [14]. A separate text processing pipeline is established for each word to be disambiguated. All these pipelines share the same preprocessing unit followed by an RCNN specialized for a single word (compare Figure 1). First, the preprocessing unit transforms a given input text containing the corresponding word into vector representations. Subsequently, these vectors are processed by the RCNN to estimate the correct sense in this context.

2.1. Vector Representation of Words

Neural Networks rely on numeric input vectors. Hence, a text must be preprocessed and transformed into such vectors. The most common methods (*bag-of-words* or *bag-of-n-grams*) lead to high dimensional and sparse representations while comprising only little semantic or contextual information (see the argumentation in [12]). In contrast, Quoc and Mikolov introduced word representations of low dimensionality preserving some semantic information [12] by mapping semantically similar words to vector representations which are close to each other in the vector space. The version of this approach used for our study is described in the following:

Let J denote the number of paragraphs in the training corpus while T_j labels the number of words in the j^{th} paragraph. The t^{th} word in the j^{th} paragraph is described by $w_{j,t}$ while p_j labels the whole j^{th} paragraph, and k represents (half) the size of a context frame of preceding and following words. The training task is to maximize (Equation 1)

$$\frac{1}{J}\sum_{j=1}^{J}\frac{1}{T_j}\sum_{t=k}^{T_j-k}\log P\left(w_{j,t}\middle|w_{j,t-k},\dots,w_{j,t-1},w_{j,t+1},\dots,w_{j,t+k},p_j\right) \tag{1}$$

The conditional probabilities P are estimated by softmax regression. At first, for every input $w_{j,t-k},\dots,w_{j,t-1},w_{j,t+1},\dots,w_{j,t+k},p_j$ the following is computed (Equation 2)

$$y = U\cdot h\left(w_{j,t-k},\dots,w_{j,t-1},w_{j,t+1},\dots,w_{j,t+k},p_j;W,D\right)+b \tag{2}$$

For this purpose, the columns of W that correspond to $w_{j,t-k},\dots,w_{j,t-1},w_{j,t+1},\dots,w_{j,t+k}$ and the column of D that corresponds to p_j are extracted and summed up. This is abbreviated by the function h in Eq. (1). The sum is linearly transformed by the weight matrix U and then translated by b. The resulting vector y has as many dimensions as there are diverse words in the training paragraphs. Assume that $y(w)$ denotes the entry of y corresponding to the word w. The required probability can then be estimated as follows (Equation 3):

$$P\left(w_{j,t}\middle|w_{j,t-k},\dots,w_{j,t-1},w_{j,t+1},\dots,w_{j,t+k},p_j\right)=\frac{e^{y\left(w_{j,t}\right)}}{\sum_i e^{y_i}} \tag{3}$$

The parameters W, D, b, U are learned via stochastic gradient descent and backpropagation. The trained network predicts a word w given the vector representations of its context words and of the paragraph it occurs in. The columns of W contain the vector representations of all words that are present in the training corpus. These representations are then used as inputs for the RCNNs.

2.2. Recurrent Convolutional Neural Networks

Disambiguation is performed by RCNNs specific for each pipeline. The RCNN proposed by Lai et al. was developed for text classification in general [13]. In our case, we use a slightly modified model for the disambiguation of words. In contrast to the original model, word embedding is done as described above. An RCNN consists of a convolutional layer with a recurrent structure, a single layer with perceptrons and a max-pooling layer followed by a final softmax layer. The input of such a network is composed of an arbitrary number of vector representations, each having fixed dimensionality. Thus, the input texts passed to the pipeline can have an arbitrary number of words. Assume that the overall input is a document $Do = w_1, \dots, w_n$ containing the word the RCNN is specialized for. During preprocessing Do is transposed into a list of vectors $e(Do) = e(w_1), \dots, e(w_n)$, which is then transferred to the RCNN. By the initial convolutional layer the following two multidimensional functions are evaluated for every $w_i \in Do$, where σ denotes the elementwise sigmoid function and $M^{(l)}, M^{(sl)}, M^{(r)}, M^{(sr)}$ are matrices (Equations 4 and 5).

$$c_l(w_i) = \sigma(M^{(l)} \cdot c_l(w_{i-1}) + M^{(sl)} \cdot e(w_{i-1})) \quad (4)$$

$$c_r(w_i) = \sigma(M^{(r)} \cdot c_r(w_{i+1}) + M^{(sr)} \cdot e(w_{i+1})) \quad (5)$$

The function c_l returns a vector representing the context defined by all words left to the current one. Analogously, c_r computes the right context. For the first input word we define $c_l(w_1)$ as a variable vector learned in addition to the other parameters. The same holds true for $c_r(w_n)$ corresponding to the last word. The whole text is considered during the context computation irrespectively of its length, which is a major advantage of a recurrent convolutional layer over a conventional convolutional layer (e.g. the approach used in [11]).

The next layer contains as many simple perceptrons as there are vector representatives. Its i^{th} perceptron computes (Equation 6)

$$y_i^{(2)} = tanh(M^{(2)} \cdot \begin{pmatrix} c_l(w_i) \\ e(w_i) \\ c_r(w_i) \end{pmatrix} + b^{(2)}) \quad (6)$$

where $M^{(2)}, b^{(2)}$ are further parameters, $\begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$ denotes the concatenation of v_1, v_2, v_3, and tanh is evaluated element wise. The subsequent max-pooling layer compresses its input consisting of arbitrary many vectors $y_i^{(2)}$ to a vector $y^{(3)}$ of fixed length by applying the max function to each vector component. The input to the softmax layer is then given by $y^{(4)} = M^{(4)} y^{(3)} + b^{(4)}$. This is transformed into to the overall output by using the softmax regression, which estimates the probability distribution of the ambiguous meanings conditioned on the input document $w_1, \dots, w_n$, or rather, on its vector representations $e(w_1), \dots, e(w_n)$.

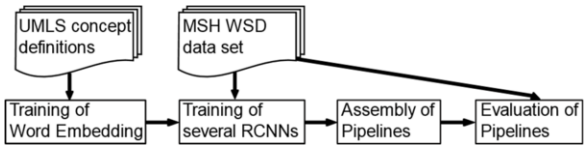

Figure 2. Line of action for the experiment

Formally, let $\theta := \{M^{(l)}, M^{(r)}, M^{(sl)}, M^{(sr)}, c_l(w_1), c_r(w_n), M^{(2)}, b^{(2)}, M^{(4)}, b^{(4)}\}$, then the probabilities are estimated by (Equation 7)

$$P(meaning\ k \mid e(w_1), \ldots, e(w_n); \theta) := \frac{e^{y_k^{(4)}}}{\sum_i e^{y_i^{(4)}}} \quad for\ all\ possible\ meanings\ k \qquad (7)$$

Training of each RCNN aims at maximizing the log-likelihood by adapting the elements of θ, i.e. at maximizing (Equation 8)

$$\frac{1}{|TrainDocs|} \sum_{Do \in TrainDocs} log\ P(correct\ meaning\ in\ context\ of\ Do \mid e(Do); \theta) \qquad (8)$$

Similar to the word embedding network, maximization is also achieved by stochastic gradient descent and backpropagation.

2.3. Experiment

All pipelines used for our studies are based on the same architecture. Documents containing an ambiguous word serve as inputs. The output of a pipeline gives an estimation of the word's correct meaning. Figure 2 depicts the line of action pursued for the experiment. The preprocessing unit is the same for all pipelines and was implemented as described in section 2.1. The training corpus for the word embedding was iterated over for 5 times with a window size of 10 words ($k = 5$) resulting in 300-dimensional word vectors.

While preserving their basic structure (see 2.2.) the trained RCNNs differ from pipeline to pipeline. While the number of columns of $M^{(4)}$ is constant for all, the number of rows equals the count of meanings of the corresponding word. In our case, the number of rows is fixed by the number of UMLS concepts linked to a given ambiguous word. The main difference, however, lies in the values of the learned parameters θ.

2.3.1. Data Set for Training and Evaluation

For the supervised training and the evaluation we needed a set of ambiguous words within context paragraphs each assigned to the UMLS concept indicating the correct meaning.

Training of the preprocessing unit for word embedding was based on the concept definition texts of the 2016AB full release of the UMLS Metathesaurus [15]. The training of the RCNNs and the evaluation of the whole pipelines were based on the MSH WSD introduced by Jimeno-Yepes et al. [8] as a benchmarking data set. MSH WSD contains 203 ambiguous abbreviations and terms. For every such entity numerous MEDLINE citations containing the term (or abbreviation) are composed in one document. Moreover, a UMLS concept is assigned to each citation, which unambiguously defines the sense of the word in this context.

20 of these documents were used to train and evaluate 20 of the previously described RCNNs. They were selected by taking the first 20 documents returned by the function os.listdir() traversing the full MSH WSD directory. According to the documentation this function returns all elements of a given directory in an arbitrary order. Thus, this subset can be considered pseudo-random. For all 20 cases the list of corresponding MEDLINE citations was divided into a training set containing about 75% of the data and a disjoint test set consisting of the rest. In both subsets the numbers of all meanings were balanced as they are in the original set [16]. Each RCNN was trained for 1000 iterations using a learning rate of 0.01. Parameters were initialized randomly based on a truncated normal distribution (mean: 0, stddev: 0.1, interval: $[-0.2, 0.2]$).

2.3.2. Technical Platform

Word embedding and RCNN training were computed on a Linux workstation (Intel Xeon E5-1650 Processor, 6-Core, 3.5 GHz; 64GB memory) using CUDA on NVIDIA GPUs (NVIDIA Quadro M4000, 1664 CUDA cores, 8GB memory). The word embedding was trained using the software framework Gensim[2] whereas the RCNN implementation is based on TensorFlow[3].

3. Results

The section presents key figures of the RCNN training process for the individual pipelines - each identified by the disambiguated word. Accuracy with respect to the test set has been measured after every 10th training iteration. An overview of these measurements is given in Figure 3.

Final accuracies after 1000 iterations are summarized in Table 1. 8 out of 20 pipelines (i.e. 40%) achieved an accuracy of 98% and above. A total of 14 out of 20 pipelines (i.e. 70%) achieved an accuracy of at least 90%. The average final accuracy of all pipelines is about 91%.

4. Discussion and Outlook

Most of the pipelines showed good or excellent performance. A clear outlier is the “Phosphorylase”-network. Its bad performance may be explained by considering the two possible meanings of the word. On the one hand, it refers to a class of enzymes and on the other hand, it describes a special enzyme of this class. Thus, one concept is enclosed by the other one (hyponymy), which makes disambiguation extremely hard.

A comparison to the Naïve Bayes approach described by Plaza et al. [6] is problematic. The Bayesian method was tested against the full MSH WSD set, but the outline of the experiment raises the suspicion that the training set overlapped the test set, which - if proved correct - would forbid to estimate the real accuracy of the proposed method by the measured one. In order to avoid these complications, we divided the MSH WSD data creating disjoint training and test sets.

[2] https://radimrehurek.com/gensim/models/doc2vec.html
[3] https://www.tensorflow.org/

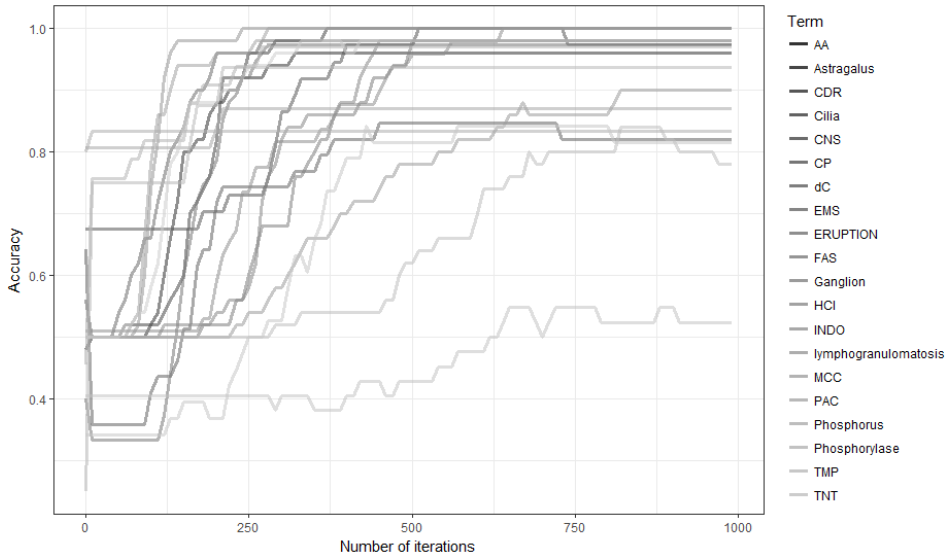

Figure 3. Summary of the intermediate accuracies

Using a general approach independent of structure or content of the UMLS promises future applicability to a broader field of biomedical content indexing.

The MSH WSD data set used for training and evaluation was assembled automatically [8]. As the labeling of data was not supervised by humans, mislabelling may obfuscate the training process. Therefore, future training data should be assembled or checked manually to avoid wrong training behaviors.

Our next aim is to dynamically combine trained RCNNs to a single general processing pipeline. The system will decide which RCNN to use for disambiguation of a given term marked as ambiguous by the UMLS. The advantage of such a system will be that new RCNNs can be added during usage. Thus, the system can continuously

Table 1. Summary of the final accuracies

Network	Number of meanings	Final Accuracy
AA	2	96%
Astragalus	2	100%
CDR	2	97%
Cilia	2	82%
CNS	2	98%
CP	3	97%
dC	2	98%
EMS	2	98%
ERUPTION	2	100%
FAS	2	100%
Ganglion	2	90%
HCl	2	100%
INDO	2	87%
lymphogranulomatosis	2	83%
MCC	2	97%
PAC	2	94%
Phosphorus	2	78%
Phosphorylase	2	52%
TMP	2	81%
TNT	2	98%

be adapted to broader WSD tasks. Feasibility of combined systems covering relevant parts of terminology is plausible based on the fact that only approximately 4 MB of checkpoint data are required to specify a trained RCNN as used in our approach. A general solution to WSD may require solutions for the most difficult problems of artificial intelligence [17]. However, a combination of solutions for small sub-problems is a good temporary solution in the meantime. The deep learning methods used by our approach recently proved very successful in other fields of text understanding and can be expected to scale-up with improved availability of training data.

References

[1] A. Kilgarriff, "I Don't Believe in Word Senses", *Computers and the Humanities*, **31**(2) (1997), 91–113.

[2] T. C. Rindflesch and A. R. Aronson, Ambiguity resolution while mapping free text to the UMLS Metathesaurus, *Proceedings of the Annual Symposium on Computer Application in Medical Care*, 1994, 240–244.

[3] A. J. Jimeno-Yepes and A. R. Aronson, Improving an automatically extracted corpus for UMLS Metathesaurus word sense disambiguation, *Procesamiento del lenguaje natural*, **45** (2010), 239–242.

[4] A. J. Yepes and A. R. Aronson, Query Expansion for UMLS Metathesaurus Disambiguation Based on Automatic Corpus Extraction, *The Ninth International Conference on Machine Learning and Applications*, 2010, 965–968.

[5] M. Stevenson, Y. Guo, and R. Gaizauskas, Acquiring sense tagged examples using relevance feedback, *Proceedings of the 22nd International Conference on Computational Linguistics*, **1** (2008), 809-816.

[6] L. Plaza, A. J. Jimeno-Yepes, A. Diaz, and A. R. Aronson, Studying the correlation between different word sense disambiguation methods and summarization effectiveness in biomedical texts, *BMC bioinformatics*, **12** (2011), 355.

[7] A. Jimeno Yepes and A. R. Aronson, Knowledge-based and knowledge-lean methods combined in unsupervised word sense disambiguation, *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*, 2012, 733.

[8] A. J. Jimeno-Yepes, B. T. McInnes, and A. R. Aronson, Exploiting MeSH indexing in MEDLINE to generate a data set for word sense disambiguation, *BMC bioinformatics*, **12** (2011), 223.

[9] D. Bahdanau, K. Cho, and Y. Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, Preprint: http://arxiv.org/pdf/1409.0473.

[10] D. Amodei et al., Deep Speech 2: End-to-End Speech Recognition in English and Mandarin, *Proceedings of The 33rd International Conference on Machine Learning*, 2016, 173–182.

[11] Y. Kim, Convolutional Neural Networks for Sentence Classification, Preprint: http://arxiv.org/pdf/1408.5882.

[12] Quoc V. Le and Tomas Mikolov, Distributed Representations of Sentences and Documents, Preprint: https://arxiv.org/pdf/1405.4053v2.pdf.

[13] S. Lai, L. Xu, K. Liu, and J. Zhao, Recurrent Convolutional Neural Networks for Text Classification, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015, 2267–2273.

[14] M. Stevenson and Y. Wilks, *Word-Sense Disambiguation*, in: The Oxford Handbook of Computational Linguistics, R. Mitkov, Oxford, 2005, 249–265.

[15] National Library of Medicine (US), *UMLS® Reference Manual*. https://www.ncbi.nlm.nih.gov/books/NBK9676/, 2009.

[16] A. J. Jimeno-Yepes and A. R. Aronson, Knowledge-based biomedical word sense disambiguation: comparison of approaches, *BMC Bioinformatics*, **11**(1) (2010), 1–12.

[17] N. Ide and J. Véronis, Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art, *Computational Linguistics*, **24**(1) (1998), 2–40.