CLRG ChemNER: A Chemical Named Entity Recognizer @ ChEMU CLEF 2020

Malarkodi C.S., Pattabhi, RK Rao., and Sobha, Lalitha Devi

Computational Linguistics Research Group,
AU-KBC Research Centre,
MIT Campus of Anna University, Chennai, India
sobha@au-kbc.org

Abstract. This paper describes our system developed for **ChEMU** @ CLEF Cheminformatics Elsevier Melbourne University lab, Named Entity Recognition (NER) task for identifying chemical compounds as well as their types in context, i.e., to assign the label of a chemical compound according to the role which the compound plays within a chemical reaction from patent documents. We have presented two systems which use Conditional random fields (CRFs) algorithms and Artificial Neural Networks (ANN). In this work we used feature set that includes linguistic, orthographical and lexical clue features. In the development of systems, we have used only the training data provided by the track organizers and no other external resources or embedding models were used. We obtained an F-score of 0.6640 using CRFs and F-Score of 0.3764 using ANN on the test data.

Keywords: Chemical named entity recognition, Artificial Neural Networks, Conditional random fields, · Patent Documents.

1 Introduction

CLEF 2020 ChEMU NER task aims to automatically identify chemical compounds and their specific types, i.e. to assign the label of a chemical compound according to the role it plays in a chemical reaction. In addition to chemical compounds, the task also requires identification of the temperature and the reaction time at which the chemical reaction was carried out, the yields obtained for the final chemical product and the label of the reaction. The focuses of this task is mainly on information extraction from chemical patents. This is a challenging task as patents are written very differently as compared to scientific literature. When writing scientific papers, authors strive to make their words as clear and straightforward as possible, whereas patent authors often seek to protect their knowledge from being fully disclosed [8]. Thus the main challenge for Natural Language Processing (NLP) in patent documents arises from its writing style,

very long winding complex sentences and listing of chemical compounds. As syntactic deep parsing is difficult for such sentence constructions, for this work we decided to use shallow parsing. The paper describes the work we have done in developing NER systems for this task "ChEMU NER task".

We pre-processed the data provided by the task organizers to the required format to develop our NER systems. Subsequently, features were extracted and trained for the identification of the entities from the corpus using Machine learning (ML) algorithms. In section 2, we briefly review the recent literature. In the following section 3, features and the method used to develop the language models are described. Results are discussed in section 4. The paper ends with the conclusion.

2 Literature Review

In recent years Deep Learning is flourishing as a well-known ML methodology for NLP applications. By using the multilayer neural architecture it can learn the hidden patterns from the enormous amount of data and handles the complex problems. This section briefly explains the recent research works in the field of NER using Deep Learning. The earlier work on neural network was done by Gallo et.al [2] to classify named entities in ungrammatical text. Their implementation of Multi-Layer Perceptron (MLP) is called as Sliding Window Neural (SwiN) which was specifically developed for grammatically problematic text where the linguistic features could fail. The Deep Neural Framework was developed by Yao et al. [11] to identify the biomedical named entities. They have trained the word representation model on PubMed database with the help of skip-gram model. Xia et al. [10] built a single neural network for identifying multi-level nested entities and non-overlapping NEs. Kuru et al.[4] used character level representation to identify named entities. They have utilized Bi-LSTMs to predict the tag distribution for each character. Wei et al.[9] have developed a CRF based neural network for identifying the disease names. Along with word embeddings their system has utilized words, PoS information, chunk information and word shape features. Hong et al.[3] developed a deep learning architecture for BioNER which is called as DTranNER. It learns the label to label transition using the contextual information. The Unary-Network concentrates on the tag-wise labelling and the pair-wise network predict the transition suitability between labels. Then these networks are plugged into the CRF of the deep learning framework. In the recent past, the models combining word level and character lever representations are being used. These methods concatenate word embeddings with LSTMs (or Bi-LSTMs) over the characters of a word, passing this representation through another sentence-level Bi-LSTM, and predicting the final tags using a final softmax or CRF layer. Lample et al. [6] introduced this architecture and achieved 85.75%, 81.74%, 90.94%, 78.76% F- scores on Spanish, Dutch, English and German NER dataset respectively from CoNLL 2002 and 2003. Dernoncourt et al. [1] implemented this model in the Neuro NER toolkit with the main goal of

providing easy usability and allowing easy plotting of real time performance and learning statistics of the model.

3 Method

In this section we present our systems developed using Condition Random Fields (CRFs) and Artificial Neural Networks (ANN). For our work we use CRF++ tool and Scikit python package. CRF++[1] tool is an open source implementation of CRFs and is a general purpose tool. Our NER system follows a pipeline architecture, where the data is first pre-processed to required format that is needed to train the system. After training the system the NEs are automatically identified from the test set.

3.1 Feature Selection

Feature selection is an important step in the ML approach for NER. Features play an important role in boosting the performance of the system. Features selected must be informative and relevant. We have used word level features, grammatical features, functional terms features that are detailed below:

1. Word level features: Word level features include Orthographical features and Morphological features.

(a) Orthographical features contain Capitalization, combination of digits, symbols and words and Greek words.

(b) Prefix and suffix of chemical entities is considered as morphological features.

2.Grammatical features: Grammatical features include word, PoS, chunks and combination of word, PoS and chunk.

3. Functional term feature: Functional term helps to identify the biological named entities and categorize them to various classes. Example: Alkyl, acid, alkanylene.

Grammatical features of Part-of-Speech (PoS) and chunk information are obtained using automatic tools. More details about the tools are given in next sub-section. The morphological features are obtained by extracting 'n' last and starting characters of chemical entities. After performing a few experiments, it was identified that n=4 is the optimal one. A Functional terms lexicon was collected from online sources.

3.2 Pre-processing

The data is pre-processed using a sentence splitter and tokenizer and is converted into column format with entities tagged using the file containing detailed chemical mention annotation (BRAT format annotation file). The sentence splitter and the tokenizer used are rule based engines, developed in-house. In these

[1] https://taku910.github.io/crfpp/

engines we have done a modification by adding a special processing to accommodate long entity names with more than 200 characters. We have split these long names into two tokens and then combined it as one after PoS tagging and Phrase chunking is completed.

Then the data is annotated for syntactic information of Part-of-speech (PoS) and Phrase Chunk information (Noun Phrase, Verb phrase) using fnTBL tool [7], an open source tool.

3.3 Named Entity Identification

The features from the pre-processed data are extracted as explained in section 3.1.The data format is similar for both the algorithms. The systems are trained using the same features extracted from the data. Using these models the chemical named entities from the test data were automatically identified. Chemical entity mention in patents requires the detection of the start and end indices corresponding to all chemical entities. Hence we converted the output from the system to the required BRAT format for task submission. The NER language models developed using CRFs, used the features as explained in section 3.1 from training. Using the NER model the NEs are automatically identified from the testing corpus. All the features were extracted from the training corpus provided by the organizers and no other external resources have been used. Similarly, we have followed for the ANN system also. ANN system is described below.

Artificial Neural Networks (ANN) A Multi-Layer Perceptron (MLP) is a feed forward Artificial Neural Network (ANN). The input layer receives the input data in the numerical form, the output layer takes the decision about the input and the hidden layers exists between these two acts as the computational engine.

The three important steps involved in neural network are 1) each input is multiplied by a weight 2) all the weighted inputs are added together with the bias and 3) the sum is passed through the activation function. The input node accepts the information in a numerical form and depending on the weight and the transfer function, the activation value which is the weighted sum of inputs will be passed to the next node. The activation function is used to monitor the threshold level and convert the unbounded value into a bounded one. Each node in the network runs the activation value and tweak the sum based on its transfer function. The activation function runs through the entire network until it reaches an output node. The traditional systems used sigmoid or the hyperbolic tangent activation function.

In this work, Multilayer Perceptron (MLP) network is used. MLP is a combination of layers of perceptrons connected together. The first layer's output goes as the input to the next layer until it reaches the output layer. The hidden layer is the layer that exists between the input and output layer Feed forward networks like MLP have two passes, namely forward and backward passes. In the forward pass, propagate forward to get the output and compare it with the

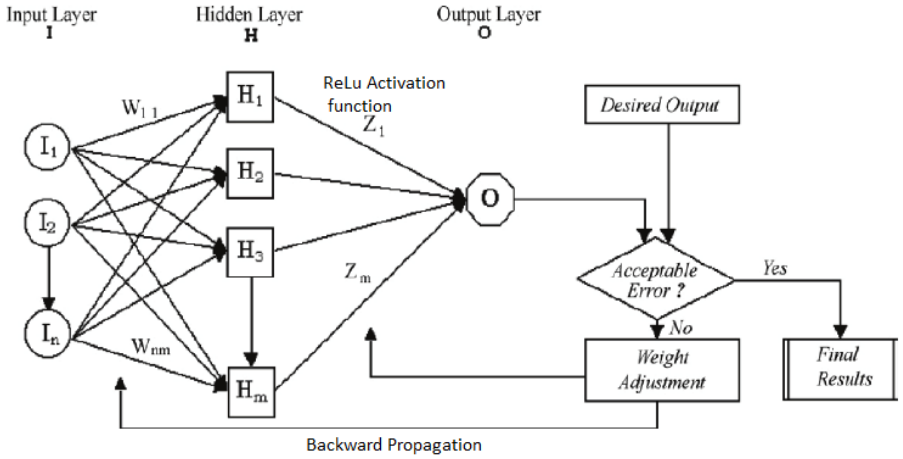

Fig. 1. ANN Architecture - of MLP Network

real value to get the error. In order to reduce the error multilayer perceptrons propagate backward and adjust the weights. This process of back-propagation is used to adjust the weight and bias relative to the error. This process continues until the estimated output is obtained. ReLu activation function is used in MLP. The feed forward network consists of two motions namely forward and backward pass. The training process comprises of 3 steps, they are forward pass, error calculation and backward pass. In forward pass the input data is multiplied by its weight and added to the bias at each node and passes through the hidden layer to the output layer. The cost function is used to predict the performance of the model, which can be computed as the difference between the predicted and the expected value. Once the loss is calculated we have to back propagate the loss in order to update the weights of each node using the gradient descent function. The weights are being tweaked according to the gradient flow in that direction. The main intention here is to minimize the loss.

In this work we have used the python package of Scikit-Learn's Multi-Layer Perceptron. The process of converting the input data into numerical feature vectors is called as vectorization and it involves mainly three steps namely tokenization, counting and normalization. The resultant data is called as Bag of words representation. In this form rather than the relative position of the words in the document the input text is represented using the word occurrences. We have used the countvectorizer to represent the data into bag of words format. It converts the text data into the numerical features. The TFIDFVectorizer is used to convert the bag of words into the matrix of TFIDF features. After initiating the size of the hidden layer and determining the activation and optimization the data is given for the training process. The ReLu activation function is used for

the hidden layers of the present MLP implementation. The stochastic gradient optimizer Adam is used for the weight optimization.

The number of layers we used for the hidden layer is 30, the activation function utilized for the hidden layer is ReLu (Rectified Linear unit function), 'adam' stochastic gradient-based optimizer is used as a solver for the weight optimization. 'Alpha' regularization parameter is set to 0.0001. The learning rate schedule used for weight updation is 'constant'. It is the constant learning rate provided by the initial learning rate which helps to control the step-size in updating the weights and the 'learning rateinit' value is set as 0.001. Batch size refers to the number of training examples used in one iteration. The batch size is assigned as mini batches for stochastic optimizers and it is set to 200 by default. The architecture of MLP implementation is shown in figure 1.

4 Results and Discussion

The system outputs on the test data were evaluated by the track organizers, precision, recall and F score were calculated. The test results are tabulated in Table 1 and 2. Table 1 provides the test results of the system developed using CRFs and Table 2 presents results of system using ANN.

Table 1. Test Results of CRFs based NER system

NE Label	Precision	Recall	F1 Score
EXAMPLE_LABEL	0.9732	0.4155	0.5823
OTHER_COMPOUND	0.9378	0.5656	0.7057
REACTION_PRODUCT	0.8600	0.4118	0.5569
REAGENT_CATALYST	0.8214	0.7495	0.7838
SOLVENT	0.8776	0.7066	0.7828
STARTING_MATERIAL	0.7192	0.7862	0.7512
TEMPERATURE	0.9802	0.7288	0.8360
TIME	0.9954	0.4779	0.6457
YIELD_OTHER	0.9315	0.1545	0.2651
YIELD_PERCENT	0.8571	0.0154	0.0303
Average	0.8793	0.5334	0.6640

The system based on CRFs had given a very good precision. The recall is low and especially for the entities "YIELD_OTHER" and "YIELD_PERCENT". This could have been improved by using post processing rules. The results obtained using ANN are lower than the CRFs based system. This clearly shows that the training data size is not sufficient for ANN. The ANN system requires used of external resources such as pre-trained word embeddings and other available annotated resources.

Table 2. Test Results of ANN based NER system

NE Label	Precision	Recall	F1 Score
EXAMPLE_LABEL	0.5455	0.0172	0.0333
OTHER_COMPOUND	0.5977	0.2430	0.3455
REACTION_PRODUCT	0.5755	0.2573	0.3556
REAGENT_CATALYST	0.8620	0.5521	0.6731
SOLVENT	0.8958	0.4066	0.5593
STARTING_MATERIAL	0.6486	0.4480	0.5299
TEMPERATURE	0.8239	0.2369	0.3680
TIME	0.9504	0.2544	0.4014
YIELD_OTHER	0.1429	0.0114	0.0211
YIELD_PERCENT	0.1429	0.0334	0.0542
Average	0.6686	0.2619	0.3764

5 Conclusion

We submitted two systems developed using Machine Learning (ML) techniques, Condition Random Fields (CRFs) and Artificial Neural Networks (ANN). A two stage pre-processing is done on development data 1) the formatting stage, where the sentence splitting, tokenizing and the data conversion to column format and 2) the data annotation stage, where the data is annotated for syntactic information of Part-of-speech (POS) and Phrase Chunk information (Noun Phrase, Verbphrase) are performed. For both POS and Chunk information, fnTBL [7], an open source tool is used. We have used the training data provided by the task organizers and have not used any external resources or pre-trained language models. The language models are developed using CRFs and ANN. The CRF++ tool is used for developing the CRF model. The ANN application uses the Scikit python package. ANN uses a Multilayer Perceptron (MLP). ReLu activation function is used in MLP. The stochastic gradient optimizer Adamis used for the weight optimization. It can adjust and calculate the learning rates for different parameters at each node. We obtained an F-score of 0.6640 using CRFs and F-Score of 0.3764 using ANN for the test data. It can be observed from the results CRFs performed better for the given training data. This shows ANN's require more training data or pre-trained models. A better solution can be arrived at by combining ANN and CRFs. In future we would like to combine ANN and CRFs.

References

1. Franck Dernoncourt, Ji Young Lee, and Peter Szolovits.: Neuroner: an easy-to-use program for named-entity recognition based on neural networks. In: arXiv preprint arXiv:1705.05487 (2017)

2. Gallo, I., Binaghi, E., Carullo, M., Lamberti, N.:Named entity recognition by neural sliding window. In:The Eighth IAPR International Workshop on Document Analysis Systems, pp. 567–573. IEEE (2008)
3. Hong, S. K., and Lee, J. G.: DTranNER: biomedical named entity recognition with deep learning-based label-label transition model. BMC Bioinformatics **21**(1), 53-73 (2020)
4. Kuru, O., Can, O. A., Yuret, D.: Charner: Character-level named entity recognition. In:Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics, pp. 911–921 (2016)
5. Lafferty, J., McCallum, A., and Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of International Conference on Machine Learning, pp. 282–289 (2001)
6. Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer.: Neural architectures for named entity recognition. In: arXiv preprint arXiv:1603.01360. (2016)
7. Grace Ngai and Radu Florian. Transformation-based learning in the fast lane. In: Proceedings of North Americal ACL 2001, pages 40–47 (2001)
8. Max, Valentinuzzi.: Patents and Scientific Papers: Quite Different Concepts. IEEE Pulse **8**(1), pp. 49-–53 (2017)
9. Q. Wei, T. Chen, R. Xu, Y. He, and L. Gui.: Disease named entity recognition by combining conditional random fields and bidirectional recurrent neural networks. Database **2016**.
10. Xia, C., Zhang, C., Yang, T., Li, Y., Du, N., Wu, X., Yu, P.: Multi-grained named entity recognition. In: arXiv preprint arXiv:1906.08449. (2019)
11. Yao, L., Liu, H., Liu, Y., Li, X., Anwar, M. W.: Biomedical named entity recognition based on deep neutral network. International Journal of Hybrid Information Technology **8**(8), pp. 279 – 288 (2015)