Vol 12 Issue 02 2023

ISSN NO: 2230-5807

Automatic Permutation Based Model for Named Entity Recognition from Nanomaterial Documents using Deep Learning Neural Network

B. Lavanya*
Department of Computer Science, University of Madras, Guindy, Chennai
lavanmu@gmail.com
G. Sasipriya
Department of Computer Science, University of Madras, Guindy, Chennai
msasipriyam@yahoo.co.in

Abstract *Due to the ever-expanding scientific publications, information extraction from them is near to impossible manually, text-mining approaches for automated information extraction are a need of the hour. Its applications include varied disciplines like biomedical data, Nanoinformatics, and other core science research articles. The properties extraction plays a significant role in the future of biomedicine. Information extraction from enormous data is the need of the hour to make headway in this field. Information search, knowledge discovery, and hypothesis development are all made easier using these strategies. The preponderance of past research has focused on improving the architecture and performance of named entity identification or relation extraction. This paper presents an extensive text mining system that combines dictionary-based entity extraction and rule-based relation extraction to develop a Permutation Based Model framework for the automatic extraction of properties and values present in the research articles. The different framework stages like extracting existing research articles content, training the NLP model, and applying the NLP model to the extracted content, are implemented in this paper. An optimized weighted average F1 score of implemented model is 0.92.*

Keywords: Automatic Extraction, Named Entity Recognition, Deep Learning, NLP

1. Introduction

Natural Language Processing (NLP) is a branch of computer science that is associated with making the computer the ability to understand and decode text as human brains do. To analyze unstructured text data, NLP employs machine learning, statistical, and deep learning models. Named Entity Recognition (NER) is a sub-branch of NLP that helps in the automatic extraction and labeling of entities.Automatic entity extraction is extremely useful when working with large datasets, and NER plays a key role in this field [1] [2].

In recent years, there is a significant increase in the field of nanoparticles, particularly in the field of nanomedicine. This led to a huge number of researchers showing interest in nanoparticles, subsequently resulting in an enormous increase in research papers especially raw data from experiments. To manually analyze all these papers are next to impossible. This emphasizes the need for automatic extraction of properties and values from the papers. The efficient way to achieve this is by using Natural Language Processing and Named entity recognition. Named entity recognition is a deep learning-based model, a semi-supervised ML algorithm trained with appropriate labels and literature. Many parameters determine the accuracy of the NER model, some of them are the quality of training data, the batch elements and size, loss, and the number of epochs. Any NER model can be optimized by tuning one or more above parameters. This paper presents a solution that extends the Open-source python NLP framework and combines the deep-learning-based NER model with rule-based parameters. Also, this solution specializes mainly in extracting the properties and values of the nanoparticles. Using only the NER model is successful to a good extent, however, the higher number of false positives degraded the efficiency of the system. Rule-based filters are added to the model, this improved the efficiency [3][4][5]. The batches are sliced using permutation-based algorithms and the results look promising compared to the most common way of compounding-based batch size.

The following section of this paper describes the study area of nanoparticles properties, Methodology, proposed architecture, Named Entity Recognition, and results and discussion.

2. Study Area

2.1. Nanoparticles properties-study

Nanoparticles are tiny particles having varying ranges in size from 1 to 100 nm [6]. Nanoparticles have been classified into different categories completely based on their size, shape, and properties [7]. Hydroxyapatite is a naturally occurring mineral obtained from calcium and phosphate. It plays a vital role as a mineral component in bones, teeth, and cancer treatments. It supports growth in bones and osteointegration, bone regeneration, hard tissue repairs, effective drug delivery systems for antibiotics [6], controlled drug-eluting systems for implant-related infections, bioimaging and diagnosis, chromatography for separation of proteins, nucleic acids, and antibodies, gene protein delivery [8], and also it is used in orthopedic and implants coated with hydroxyapatite inmaxillofacial application [9][10]. Some other applications of Hap are commercial cosmetics, sporting, goods, sunscreen, drug/gene/protein carriers, etc., [11].

2.2.1. Particle description

The corpus was constructed for the experimental information extraction for the nanoparticle, the corpus was constructed from the full text of research papers. Some of the corpora are listed below with definitions and examples [4].

Physical properties: Physical properties of nanoparticles are weight, shape, color, temperature, density, etc., by using physical properties we can predict the size of the nanoparticle.
Manufacturing and experimental techniques: It briefly describes the preparation methods like Scanning Electron Microscopy (SEM), dynamic light scattering (DLS), Fourier Transform Infrared Spectroscopy (FTIR), laser light scattering (LLS), Raman Spectroscopy, laser light diffraction, Thermal Analysis, X-ray diffraction Differential thermal analysis (DTA), X-ray photoelectron spectroscopy, thermal gravimetric analysis (TGA), etc., [12].
Morphology: It has special importance to nanoparticles because nanoparticles are strongly correlated with various shapes like rods, crystals, spherical, etc., [12].
Chemical properties: Chemical properties that are important in characterizing materials include composition, structure, molecular weight, boiling and melting points, vapor pressure, water solubility, reactivity, stability, etc.,
Chemical composition: The chemical composition of nanoparticles can range from single raw material to multiple ones, for example in hydroxyapatite calcium and phosphate, is the main substitute, and the value ranges from 1.6 to 1.7.
Mechanical properties: It is critical to understand the mechanical characteristics of a materialbefore deciding on it for an engineering product or application. A material's mechanical characteristics are those that influence its mechanical strength and ability to be molded into the desiredshape. The following are some of a material's typical mechanical properties: Strength, Toughness, Hardness, Hardenability, Brittleness, Malleability, Ductility, Creep and Slip, Resilience, and Fatigue. As for traditional materials, the mechanical properties of hydroxyapatite generally consist ofYoung's Modulus, Weibull's Modulus, and Vicker's hardness [13].

3. Entity extraction-nanoparticle

NER (Named Entity Recognition) techniques in the field of nano informatics have been categorized into different approaches: machine learning-based, rule-based, dictionary-based, ontology-based, and hybrid-based. Only very little of the work has been developed in the field of nano informatics, initially working on rules, POS (part-of-speech), and simple dependencies. Before using these different approaches, entity recognition extraction is done manually, which means annotating by domain experts.
An annotation framework for extracting the information from research articles has been proposed in this work [14]. Then experimental parameter extraction can be done by discussing with the nanodevice domain expert, evaluation has been done by using the kappa coefficient with the tight and loose agreement of precision, recall, and f1 scoresof 0.89%, 0.76%, 0.94%, 0.80%.
In the recent corpus creation for the high quality of data pro gene [15], they have listed the corpus into 11 subcategories from various biological domains. Two annotators carefully assigned the entities mentioned to the genes/proteins and their families into a single class. And also worked with flair and bio-Bert with an accuracy of 97.59% and 97.04%, Precision- 84.66% and 79.77%, Recall- 85.39% and 81.21%, F1- 0.850%, 0.805% and it is publicly available protein/gene corpus across the world. In the performance, score flair gives a slightly better result compared to bio-Bert.

The paragraph-based approach [16] has been used in place of a sentence-based approach for relation extraction. Also, they extracted relations among all of the nine entities and attributes defined in the nanotoxicity ontology in place of extracting binary relations between two entities. Named entity recognition for this nanotoxicity information extraction had been done by using ontology-based entity and relation extraction with precision-82.9% recall-79.7%, and f-measure-81.3%. The performance evaluation has been compared by using gold standards.
The automatic extraction to acquire PAMAM Dendrimer properties [17] for the cancer treatment literature with precision - 0.84%, recall- 0.99%, and f1measure - 0.99%. The values are taken from the NanoParticle Ontology, which is a polymeric nanoparticle, and highly branched that can be easily modified to different specifications.
By comparing two state-of-the-art named substance extraction frameworks, OpenNLP and Stanford NER, this study analyses the trustworthiness of the manual comments and illustrates the use of the corpus. The clarified corpus is accessible open-source and, based on these comes about, rules and proposals for future development of extra nanomedicine corpora are given. This paper validated our system by clarifying a corpus of FDA-approved nanomedicines from sedate item embeds collected from the Drugs FDA Database. And, annotated the nanoparticle physicochemical properties, presentation parameters, and biological reaction data and assessed the unwavering quality of the human appraisals, with precision- 0.92%, recall- 0.90%, and F-measure- 0.91%. We have compared the entity extraction with a new framework for a nanoparticle property and its value, with good precision and recall.

4. Methodologies

4.1. System architecture

Figure (1) represents the overall system architecture of our information extraction on nanoparticles mainly focusing on the SpaCy framework offers a variety of text processing natural language processing tools, such as tokenization, sentence splitting, POS tagging, lemmatization, and dependency parser. Followed by tree-based rules to the extracted sentences and the relation of property and value [16][3].
The first step in this work is to extract the research papers that are available in PDF format, then extract the entities. Training the data, building a corpus, labeling for tags, and subsequently performing NLP (Natural Language Processing) for the extracted sentence are all part of this crucial step.

4.2. Data Preprocessing

Data pre-processing is the initial stage in creating an NLP model. The text data in raw form is used, which means it may contain numerous mistakes as well as a lot of unwanted content, causing our results to be inaccurate. As a result, pre-processing our data and making it easier to comprehend and analyze is required to get better results.

4.2.1. Data from the literature

PDF is the most preferred format for storing research papers in the field of nanoparticles. There is a huge number of articles present in this format. In the proposed work, the information is extracted from the unstructured text in the PDF literature. The first is going to extract the data from the research articles, that is information extraction from the PDF while extracting information from PDF needs to clean references, bibliography, title, and abstract.

4.2.2. Regex cleaning

The sentences are cleaned using regex patterns to clean page headers, footers, web page URIs, mail addresses, dates and years, and reference citations to make the document clean with only necessary research information.

4.2.3. PDF and Sentence extraction

PDF is one of the most unstructured text data file formats. To extract information from the files, all the text found in the file is extracted. The first step to extracting the information from unstructured text is to tokenize them into sentences. Open-Source python framework is used to tokenize the sentences using correct punctuations.

4.3. Preprocessing

The preprocessing involves the following sub-tasks like Sentence and Word Tokenization, Punctuation, POS Tagging, Dependency parsing, and Abbreviation expansion.

4.3.1. Sentence and Word tokenization

In every NLP framework, the first and most essential step is tokenization. To begin, the unstructured text data must be tokenized into sentences. This may be accomplished by using suitable punctuation. Tokenization transforms a word into a machine-understandable format. The sentences are tokenized using appropriate tokens like space, punctuation, etc. using tokenizers. These symbols aid in comprehending the meaning of a word in a phrase [18].

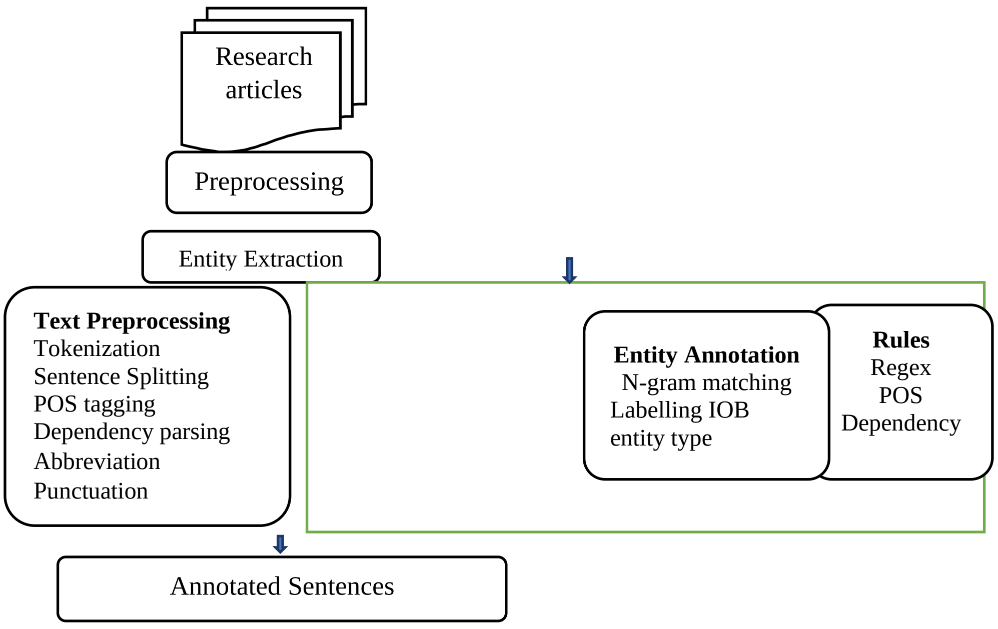

Figure 1: Overall Proposed System Architecture for Entity Extraction

4.3.2. POS tagging and Dependency Parsing

The next task is the Part of Speech (POS) Tagging, which involves assigning grammatical tags to every token. As the name indicates, they involve tagging words as nouns, pronouns, verbs, adverbs, adjectives, conjunction, and so on. This process is more important in understanding the context of the words used. They define the characteristic structure of lexical terms used in a sentence. Penn Treebank POS tags for tagging, are used to extract POS tags.
Dependency parsing involves finding the root word for each sentence and relating each word with the root word to form a tree-like structure with a parent-child relationship. Also called syntactic parsing, they detect the syntactic structure of the sentence. This is more important in the relation extraction of the entities. Noun chunks are sequences of words containing one or more nouns. Extracting the noun chunks plays an important role in dependency parsing.

4.3.3. Abbreviation expansion and Punctation

Science sentences have so many common abbreviations. If not resolved correctly, these entities will not be predicted. This will increase the false negatives. The abbreviations are resolved using the Abbreviation Detector pipe of the python-based open-source framework [19].
Scientific articles contain several standard abbreviations. If not resolved correctly, these entities will not be predicted. This will increase the false negatives. Domain-specific dictionaries help resolve these abbreviations [19].

4.4. Entity annotation

Annotating entities through NER is an important subtask of NLP. Here the system predicts entities and labels predefined categories. The trained model is capable of detecting entities for property and values for nanoparticles
N-gram matching:
In linguistics, an n-gram is an adjoining sequence of multiple words in a sentence. The important usage of n-gram is Appropriate String matching. The skip grams can be used if not all the entities are in a contiguous sequence.

Vol 12 Issue 02 2023

ISSN NO: 2230-5807

Labeling IOB entity type
The IOB scheme is one of the two labeling schemes for multi-token labels. The tokens are divided into three categories, B for the Beginning of a multi-token entity or a single token entity. I for inside a multi-token entity and O for outside an entity.

Rules
The true positives in our entity prediction using NER are more promising, but there are issues with the false negatives. One way of reducing false positives is through rule-based filtering [18].

Regex: These false negatives can be solved by filtering the entities with regex-based rules. Regex patterns help in removing sentences without stop words. Some sentences like table headers and Figure names can have entities but are not meaningful enough to make it into the final list.

POS and Dependency:
The property values of nanoparticles have numbers in them. The numerals in the headers list must be distinguished and removed, as they may cause false positives. The numerals present in the headings need to be removed as these may lead to false positives. These numerals need to be found and filtered, so that they are mean values based on the POS tags like NUM, NOUN, and PNOUN, and dependency tags like num mod, compound, and appos(appositional modifier).

5. Named entity extraction

5.1. Generating the Train Data

The input for train data is given as input which includes the sentence to be trained with details of entities and labels. XMLs will be easier to expand and maintain for future usage. The content from the XML files is parsed and converted into a system-understandable format. Figure 2 represents the next essential step of this work, how the data is trained, building the corpus, NER labeling for compounding, and permutations [20].

5.2. Building corpus

The corpus is one of the most important parts of any NLP system. In most cases, larger corpus data increases the efficiency of the system [21]. But not necessarily in all cases, as too much data may slow the performance of the system. Choosing high-quality unambiguous data improves the performance of the system.

5.3. NER Labelling

After creating the corpus, the labels for entities are added to the NER pipeline of the model. Some of the labels added are physical properties, chemical properties, chemical composition, experimental techniques, manufacturing technique, morphology, mechanical properties, property value, etc.,

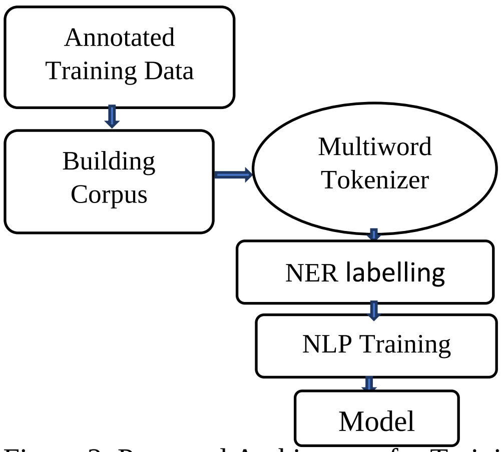

Figure 2: Proposed Architecture for Training Data

5.4. Training with Compounding and Permutation-based Methods

Controlling batch size is one of the important factors in the efficient optimization of deep learning [22]. Using python open-source libraries, two different approaches are implemented to train the data. One was the most common implementation of using minibatch with compounding provided where the batches are sliced using compounding. Here the Optimization is done once every batch.
As an alternative, the permutation-based algorithm is used for slicing batch sizes. The random permutation is used for shuffling and slicing the train data. Here the Optimization is done for every batch through SGD (Stochastic Gradient Descent). Both the models are trained using the open-source python NLP framework by the deep learning algorithm. Then they are tested and stored as model files. The models are now complete and ready to be used.

5.5. Permutations-Based-Model

This algorithm1 is a new approach to generating random mini-batches. The XML format is utilized for the input training data. The training sentences, entities to train, and labels for the entities are all provided in the XML. As an alternative, the model might be trained using batches that were randomly permutated and sliced into mini-batches. Here the optimization is done for every minibatch randomly. The results from this model look promising. Inalgorithm1 where p represents the permutation, z represents the train data, e represents epoch, m represents the length of the train data, and r represents the size of random mini-batches.

```
Input:m, b, p, z, e, r/
Output:sentences random batches
FunctionRandom_Mini_Batch (z, r)
  m ← len(z)
    p ← list(p(m))
    Call shuffle data(z, p)
  B ← (m/r)
  for k ← 0 to r do
    $b_1, b_2, b_3, \dots, b_r = p_k((k*r),(k+1)*r)$
  if m % r >0 then
    $b_{r+1}$ ← append.(($p_{k+1}$(b*r),m)
    return b
Function shuffle data (z, p)
  for i 1 to ← n do
    Shuffle list. append($z_i$)
  return shuffled list
Function Main ()
  for every epoch e do
        Call Random Mini Batch (z, r)
  for b in batches do
  for txt in b do
    NLP.Update()
  return model
```

Algorithm 1: Permutation Based Model

6. Result and Discussion

6.1. System Implementation

The models have been tested in Microsoft Windows 10 64-bit Operating System with Intel Core i5-1035G4 Processor (Core i5-1035G4) with integrated GPU, 8GB DDR4-2667, and 256 GB PCIeNVMe SSD. The models are run in Jupyter Notebook with python. Both permutation and compounding-based models run with custom-trained NER models, using python libraries [23].
The training data used is around, and the parsed data from XML is given as input for model training. There are a total of 2594 sentences extracted from the research publications. After cleaning headers, footers, email addresses, and references, the number of sentences relevant for properties extraction is 901. The number of entities predicted

using the permutation-based model is 718 and for the compounding-based model is 1139. After applying rule-based filters the entity count was reduced to 327 and 429 respectively. The number of unique entities predicted is 265 and 362 respectively.
The trained model is capable of detecting entities for property and values for nanoparticles. The typical training parameters utilized in this implementation (batch size – 2, number of epochs – 40, dropout – 0.2) were improved, resulting in an efficient model. In the compounding-based model, the batch sizes are sliced using the compounding method which compounds by 1.001 with a maximum batch size of 32. In a permutation-based model, the training data is shuffled using random permutations and the data is ordered based on the permutation. Then they are sliced at a predefined batch size of 2. The time taken for model creation is significantly lesser in the permutation-based model. The resultant models are saved and these saved models are loaded while extracting the entities.
Conditional Random Field (CRF) is the most commonly used statistical model in Named entity recognition mainly for sequential predictions. By using CRF the results are validated by splitting train and test data in ratios of 90 and 10 with 10-fold cross-validation [24][25]. The tagging results are discussed in the format of the IOB tag. The algorithm used LBFGS with a maximum number of iterations of 40 for both permutation and compounding. The results of the permutation-based model are better than the compounding-based model. The weighted average F1-Score of the permutation-based model is 0.918 against 0.86 for a compounding-based model. By using this statistical measure, the results for the permutation-based algorithm are better than the compounding-based, the results are discussed in detail in the tables.

6.2. Performance Evaluation

The chart shown in Figure.3, gives the number of entities predicted by both compounding and permutation. The PHYSICAL PROPERTIES and PROPERTY VALUE labels have more values predicted in the compounding-based model.

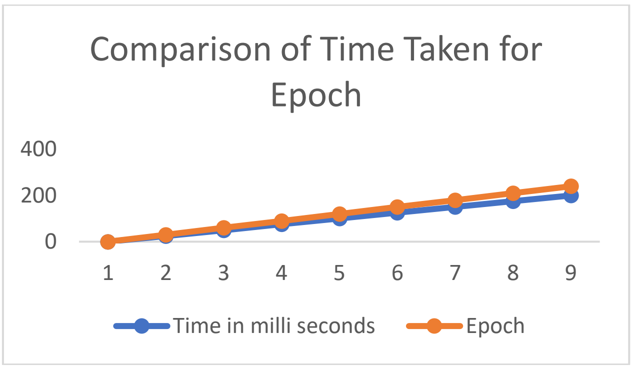

Figure 4: Comparison of Time Taken for Epoch

For further analysis, the precision values are evaluated. The precision values of PHYSICAL PROPERTIES are 0.37037 and 0.615385 and for compounding-based and permutation-based models respectively. The precision of PROPERTY VALUE is 0.794872 and 0.891892. This shows the permutation-based model has lesser false positives and hence performs better in automatic entity labeling.

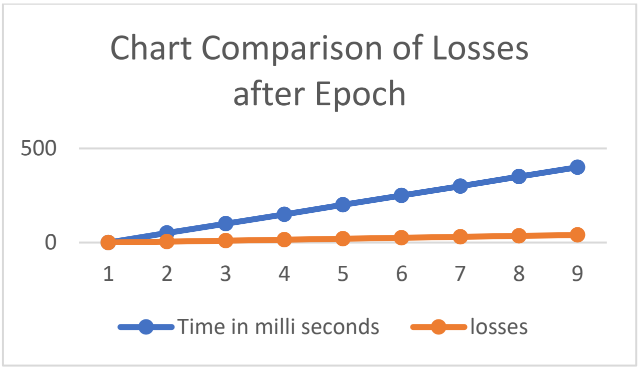

Figure 5: Comparison of Losses after Epoch

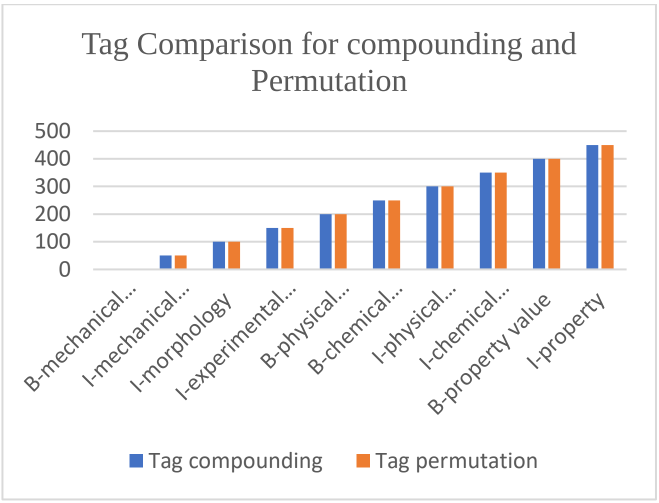

Figure 3: Tag Comparison for compounding and Permutation

It is always preferable to have a low loss function value. Losses are usually calculated and adjusted at the end of each epoch, where an epoch is a complete iteration of the whole training dataset. In the permutation-based model, the loss value for the best model (9.3) is 20% lesser than the compounding-based model (11.6) as illustrated in figure 4.
Efficient use of time and hardware is the most important aspect of a good software solution. The permutation-based model consumes 45% lesser time for 40 epochs (110.432 seconds) to run compared to the compounding-based model (197.991 seconds). The comparison of the time between the models is shown in the following figure 5.
The following sentence illustrates the entity representation for the sentences:
Sentence 1: The powder particles are polycrystalline with an average size of 5-90 nm.
Sentence 2: The bending, compressive, and tensile strength values of HA ceramics lie in the range of 30-250, 120-150, and 38-300 Mpa respectively.
As per the result "Polycrystalline as Physical Property and 5-90nm as value", "The bending, compressive, and tensile" as Mechanical Property" and "30-250, 120-150 and 38-300 Mpa" as values. However, in the Mechanical Property, multiple entities are presented in a sentence, according to training data the entities have been extracted individually.

6.3. Performance Validity Measure

For any software or framework, the desired result is which takes it to the next level. Universally accepted performance measures precision(P), recall(R), and F1-Score(F) are used for evaluating these models [20]. While the precision score will be better with a decrease in false positives, the recall score will increase with a decrease in false negatives. The harmonic mean of accuracy and recall is used to calculate the F1-Score. The formula to calculate precision and recall is classified.

Vol 12 Issue 02 2023

ISSN NO: 2230-5807

$$Precision = \frac{TP}{TP + FP}$$
$$Recall = \frac{TP}{TP + FN}$$
$$F1\ Score = \frac{2PR}{Precision + Recall}$$
$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$
$$Error\ Rate = \frac{FP + FN}{TP + TN + FN + FP}$$

True Positive (TP)-If both the predicted output values and the gold standard values are true.**False Positive (FP)-**If the predicted value is true, but the value is false in the gold standard.**True Negative (TN)-**If both the predicted output and the gold standard values are false.**False Negative (FN)-**If the predicted value is false, but it is true in the gold standard.**Accuracy-**Percentage of correctly predicted information extraction by the machine.**Error Rate-**Percentage of not correctly predicted information by the machine.
From the results calculated, the micro, and macro F1-score of a permutation-based model (0.84 and 0.885405) is better than the compounding-based model (0.771429 and 0.786923). In this experiment, a confusion matrix is used to measure the accuracy of true positive, true negative, false positive, and false negative. The scores are discussed in detail in the tables. The table1 represents the comparison between permutation and compounding, the predicted values with the gold standards.
Table 2 represents the comparison between compounding and permutation with the statistical measure CRF, and calculated precision, recall, and f1- score. By using these values able to find the accuracy for compounding is 0.86, for permutation 0.906, the error rate for compounding is 0.13 and for permutation is 0.096 for the extracted values [18].

6.4. Time Complexity

The Permutation based algorithm's computing complexity is determined as follows, Time Complexity for the permutation-based algorithm is O((m+(m/r)) *n) at its best case and its average case. Here m represents the length of train data and r

Labels	**Methods**	**Precision**	**Recall**	**F1 score**
Mineral	permutation compounding	1 1	1 1	1 1
chemical composition	permutation compounding	0.667 0.5	0.8 0.6	0.727 0.546
Property value	permutation compounding	0.9 0.9	0.875 0.875	0.889 0.889
physical Properties	permutation compounding	0.6 0.4	0.9 0.889	0.72 0.551
Experimental Technique	permutation compounding	0.667 0.5	1 0.5	0.8 0.5
Mechanical Properties	permutation compounding	1 0.889	1 0.889	1 0.889
Bone Composition	permutation compounding	1 0.833	1 1	1 0.909
Micro Average	permutation compounding	0.805 0.71	0.892 0.843	0.846 0.771
Macro Avg	permutation compounding	0.939 0.821	0.876 0.754	0.906 0.786

Table 1: Performance evaluation compared with Gold Standards

Labels	Methods	Precision	Recall	F1 score
O	permutation compounding	0.94 0.897	0.989 0.987	0.964 0.94
B-chemical composition	permutation compounding	0.538 0.917	0.318 0.355	0.4 0.512
I-chemical composition	permutation compounding	0.357 0.88	0.326 0.256	0.341 0.396
B-chemical properties	permutation compounding	1 1	0.3 0.286	0.462 0.44
I-chemical properties	permutation compounding	1 1	0.5 0.2	0.667 0.333
B-experimental technique	permutation compounding	0.833 1	0.278 0.25	0.417 0.4
I-experimental technique	permutation compounding	0.833 1	0.217 0.154	0.345 0.267
Bmanufacture technique	permutation compounding	1 0.75	0.429 0.6	0.6 0.667
I-manufacture technique	permutation compounding	1 0.75	0.429 0.6	0.6 0.667
B-property value	permutation compounding	0.919 0.774	0.687 0.6	0.786 0.676
I-property value	permutation compounding	0.933 0.964	0.764 0.474	0.84 0.635
B-physical properties	permutation compounding	1 0.471	0.034 0.113	0.067 0.182
I-physical properties	permutation compounding	1 0.276	0.042 0.082	0.08 0.127
B-mechanical properties	permutation compounding	0.714 1	0.294 0.182	0.417 0.308
I- mechanical properties	permutation compounding	0.714 1	0.25 0.25	0.37 0.4
Micro Average	permutation compounding	0.932 0.888	0.932 0.888	0.932 0.888
Macro Avg	permutation compounding	0.529 0.513	0.231 0.212	0.295 0.275
Weighted Avg	permutation compounding	0.926 0.865	0.932 0.888	0.918 0.86

Table 2: CRF-performance evaluation for entity Extraction

6.4. Time Complexity

The Permutation based algorithm's computing complexity is determined as follows, Time Complexity for the permutation-based algorithm is O((m+(m/r)) *n) at its best case and its average case. Here m represents the length of train data, and r represents the number of items in a batch. This step is performed for 'm+(m/r)' the number of times every epoch (for 'm' and 'r') and r = 1,2, 3, . . . n. Here update happens for each batch in one iteration therefore, the total number of basic steps performed, or the complexity is given: [O((m + (m/r))+O(m)] *n), where n represents the number of epochs. The worst-case scenario depends on the size of the random mini-batches.Therefore, at best case and average case of the Permutation based algorithm = [O((m + (m/r))+O(m)] *n).

7. Conclusion

The entity extraction system developed by us is more accurate than existing solutions. The F1 score achieved using permutations (0.92) is better than the minibatch compounding approach (0.88). Also, the weighted average CRF F1 score using permutations (0.9188) is better than the minibatch compounding approach (0.88). Our proposed solution can also act as middleware for applications. As of now, only a few research papers are extracted using our entity extraction solution in the biomedical properties extraction field. It is our goal to expand it to include a large number of research publications. Anyone with little computer knowledge can easily give input data in XML files. The system may get slow for a large number of files. The performance concerns in the solution for production data deployment are our next priority. This framework is the first step towards developing a complete named entity extraction solution for biomedical properties extraction.

Acknowledgment

This research work was supported by DST-RUSA through the University of Madras under the RUSA 2.0 project.

References

[1] V. K. Vedantam, the survey: Advances in natural language processing using deep learning, Turkish Journal of Computer and Mathematics Education (TURCOMAT) 12 (4) (2021) 1035–1040.
[2] B. Bhasuran, J. Natarajan, Automatic extraction of gene-disease associations from literature using joint ensemble learning, PloS one 13 (7) (2018) e0200699.
[3] M. Song, W. C. Kim, D. Lee, G. E. Heo, K. Y. Kang, Pkde4j: Entity and relation extraction for public knowledge discovery, Journal of biomedical informatics 57 (2015) 320–332.
[4] N. Perera, M. Dehmer, F. Emmert-Streib, Named entity recognition and relation detection for biomedical information extraction, Frontiers in Cell and Developmental Biology 8 (2020) 673.
[5] I. Charles, Machine learning: Data extraction technique using na¨ıveBayes algorithm.
[6] A. Gavaskar, D. Rojas, F. Videla, Nanotechnology: the scope and potential
applications in orthopedic surgery, European Journal of OrthopaedicSurgery & Traumatology 28 (7) (2018) 1257–1260.
[7] I. Khan, K. Saeed, I. Khan, Nanoparticles: Properties, applications, and toxicities, Arabian journal of chemistry 12 (7) (2019) 908–931.
[8] K. Lin, J. Chang, Structure and properties of hydroxyapatite for biomedical applications, in Hydroxyapatite (HAp) for biomedical applications, Elsevier, 2015, pp. 3–19.
[9] R. Petit, The use of hydroxyapatite in orthopedic surgery: a ten-year review, European Journal of Orthopaedic Surgery & Traumatology 9 (2) (1999) 71–74.
[10] H. Oonishi, H. Oonishi, H. Ohashi, I. Kawahara, Y. Hanaoka, R. Iwata, L. L. Hench, Clinical applications of hydroxyapatite in orthopedics, in: Advances in calcium phosphate biomaterials, Springer, 2014, pp. 19–49.
[11] J. Jeevanandam, A. Barhoum, Y. S. Chan, A. Dufresne, M. K. Danquah, Review on nanoparticles and nanostructured materials: history, sources, toxicity and regulations, Beilstein journal of nanotechnology 9 (1) (2018) 1050–1074.
[12] M. Sanyal, A. Datta, S. Hazra, Morphology of nanostructured materials, Pure and Applied Chemistry 74 (9) (2002) 1553–1570.
[13] D. Obada, E. Dauda, J. Abifarin, D. Dodoo-Arhin, N. Bansod, Mechanical properties of natural hydroxyapatite using low cold compaction pressure:Effect of sintering temperature, Materials Chemistry and Physics 239(2020) 122099.
[14] T. M. Dieb, M. Yoshioka, S. Hara, M. C. Newton, Framework for automaticinformation extraction from research papers on nanocrystal devices,Beilstein journal of nanotechnology 6 (1) (2015) 1872–1882.

[15] E. Faessler, L. Modersohn, C. Lohr, U. Hahn, Progene-a large-scale, high-quality protein-gene annotated benchmark corpus, in Proceedings of the 12th Language Resources and Evaluation Conference, 2020, pp. 4585–4596.
[16] L. Xiao, K. Tang, X. Liu, H. Yang, Z. Chen, R. Xu, Information extractionfrom nanotoxicity related publications, in 2013 IEEE International Conference on Bioinformatics and Biomedicine, IEEE, 2013, pp. 25–30.
[17] D. E. Jones, S. Igo, J. Hurdle, J. C. Facelli, Automatic extraction of nanoparticleproperties using natural language processing: Nanosifter an application to acquire paramdendrimer properties, PLoS One 9 (1) (2014)e83932.

[18] A. M. Aubaid, A. Mishra, A rule-based approach to embedding techniquesfor text document classification, Applied Sciences 10 (11) (2020) 4009.
[19] M. Neumann, D. King, I. Beltagy, W. Ammar, Scispacy: Fast and robustmodels for biomedical natural language processing, arXiv preprint arXiv:1902.07669 (2019).
[20] T. M. Dieb, M. Yoshioka, S. Hara, Automatic information extraction ofexperiments from nanodevices development papers, in 2012 IIAI InternationalConference on Advanced Applied Informatics, IEEE, 2012, pp.42–47.
[21] S. A. Salloum, M. Al-Emran, A. A. Monem, K. Shaalan, Using text mining techniques for extracting information from research articles, in Intelligentnatural language processing: Trends and Applications, Springer, 2018, pp.373–397.
[22] F. He, T. Liu, D. Tao, Control batch size and learning rate to generalizewell: Theoretical and empirical evidence (2019).
[23] X. Schmitt, S. Kubler, J. Robert, M. Papadakis, Y. LeTraon, A replicablecomparison study of ner software: Stanfordnlp, nltk, opennlp, spacy, gate, in 2019 Sixth International Conference on Social Networks Analysis,Management and Security (SNAMS), IEEE, 2019, pp. 338–343.
[24] M. Dias, J. Bone, J. C. Ferreira, R. Ribeiro, R. Maia, Named entity recognitionfor sensitive data discovery in Portuguese, Applied Sciences 10 (7) (2020) 2303.
[25] M. Garc´ıa-Remesal, A. Garc´ıa-Ruiz, D. Perez-Rey, D. De La Iglesia,V. Maojo, Using nanoinformatics methods for automatically identifyingrelevant nanotoxicology entities from the literature, BioMed research international
2013 (2013).