THE LISTER HILL NATIONAL CENTER
FOR BIOMEDICAL COMMUNICATIONS
A research division of the U.S. National Library of Medicine

A report to the Applied Clinical Informatics Branch

Check Tags MeSH Terms Indexing Research Project

May 2023

Daniel Le, Ph.D.

James G. Mork, M.Sc.

U.S. National Library of Medicine, LHNCBC
8600 Rockville Pike, Building 38A
Bethesda, MD 20894

CHECK TAGS MESH TERMS (CTMTS) INDEXING PROJECT

1. Introduction

MEDLINE® is the largest bibliographic database of life sciences and biomedical information created and maintained by the National Library of Medicine (NLM). The database contains over 30 million citations indexed with NLM Medical Subject Headings (MeSH®).

MEDLINE documents are indexed using about 30,000 MeSH terms by the NLM. Among these MeSH terms, a small subset of 40 most frequently indexed MeSH terms known as Check Tags help identify age groups, human or animal, males or females, historical periods, and pregnancy that are mentioned in almost every article.

This project describes an ongoing effort at the NLM to automate the indexing of 40 Check Tags MeSH terms (CTMTs) based on titles and abstracts in the MEDLINE literature using various techniques and algorithms in Deep Learning, Ensemble Random Forest Bagging Machine Learning, and Natural Language Processing.

Over the years, MeSH indexing for MEDLINE was done mostly by highly trained human indexers who read the full text of journal articles and assign appropriate MeSH terms to the articles. In April 2022, NLM decided to go with the full automation of indexing for all journals indexed for MEDLINE. The automated indexing of MEDLINE citations "provides users with timely access to MeSH indexing metadata and allow NLM to scale MeSH indexing for MEDLINE to the increasing volume of published biomedical literature" [1]. In the recent NLM MeSH indexing for MEDLINE report, it showed that the automated system had resolved backlogs of citations needing to be indexed in MEDLINE, reduced the cost of indexing, and can add MeSH indexing to articles within 24 hours.

A distinctive feature of using MeSH terms to search is that users can find all articles related to MeSH terms' concepts, regardless of the terms or words used in the articles. This is different from Internet search engines like Google, Microsoft Bing, or Yahoo, which search based on the same words.

Automated indexing of MEDLINE citations with MeSH terms is a challenging multi-label classification problem due to the large number of labels (MeSH Headings) and very imbalanced datasets. Regarding data used between manual indexing and automatic methods for assigning MeSH terms, there is another challenging problem where NLM human indexers have access to the full text while automated indexing methods only use title and abstract [2].

Several studies addressing these challenging problems have been reported in the Natural Language Processing (NLP) literature. For example, the well-known Medical Text Indexer (MTI) [3] machine learning system developed by NLM is a rule-based automated indexing system that processes an article title and abstract and recommends MeSH terms to human indexers. MetaLabeler [4] used the MetaLabeler algorithm proposed by Tang et al. [5] to handle the MeSH indexing challenge. MeSH Now [6], MeSHLabeler [7] and DeepMeSH [8] incorporated the learning-to-rank approaches for improving the results of automatic MeSH indexing. Recently, due to the popularity of Deep Learning, AttentionMeSH [9], "Convolutional Neural Network for Automatic MeSH Indexing" [10], and MeSHProbeNet [11] were designed based on the Deep Learning neural network multi-label classification approaches for automatic MeSH indexing.

In this manuscript, we describe a project to index 40 Check Tag MeSH terms using Deep Learning neural network multi-label classifiers, followed by Random Forest Bagging machine learning classifiers that combine predictions from multiple neural network classifiers to improve the system's predictive performance. Note that for this work, the total number of labels is 40 CTMTs, but the other two challenges remain since the distributions of their CTMTs are very highly imbalanced, and the automated indexing methods can only access titles and abstracts. The features used for the Deep Learning neural networks are the combinations of open-source documents/sentences embeddings vectors and the project customized vectors. The open-source embeddings vectors include Universal Sentences Encoder vectors, Sentence Transformers Embeddings vectors, and Biomedical Sentence Embeddings vectors. The customized vectors consist of MeSH entry terms-based vectors, and word-based dictionary vectors. All vectors are generated from titles and abstracts of MEDLINE documents.

Experiments conducted on several million MEDLINE citations show that our proposed approach, which is based on a two-level chained method of Deep Learning neural networks classifiers and Random Forest Bagging machine learning classifiers, has a competitive performance with 86.0% precision, 81.1% recall, and 83.5% F1-score.

Note that in this manuscript, the two words "documents" and "articles" have the same meaning, so they can be used interchangeably.

2. Project Objectives

The overall goal of the project is to help improve the performance of automated MeSH indexing for MEDLINE citations and thereby control their cost. This goal could be achieved through the following specific objectives:

Objective 1: Research and develop advanced machine-learning techniques for automated indexing of biomedical articles with Check Tags MeSH terms.

We have been conducting research and development to design, develop, and evaluate advanced machine-learning techniques to assist human indexers with recommended MeSH terms for indexing articles as well as to automatically generate MeSH terms for selected groups of biomedical articles.

In particular, the research and development in this active research area includes the Medical Text Indexer (MTI). MTI is a rule-based automated indexing system developed by NLM that processes an article title and abstract and returns a list of recommended MeSH terms.

Objective 2: Build a practical production system and use it as an experimental test bed to conduct research in automated indexing of biomedical articles with Check Tags MeSH terms and to identify opportunities for improving system performance.

3. Project Significance

The outcome of this project is planned to replace a current machine learning software component of the NLM Medical Text Indexer (MTI) system that classifies Check Tags MeSH terms (CTMTs) using ruled-based algorithms.

In addition, this project proposed an interesting architecture to handle multi-label classification problems that provides a new perspective by:

- Combining embedded and customized feature vectors as its inputs.
- Merging datasets and their spin-off datasets to improve predictive performance.
- Chaining two different machine learning classification approaches: Deep Learning neural networks and Random Forest Bagging machine learning models to index CTMTs.

As a result, this proposed architecture could set up a platform for possible future experiments to improve the performance of CTMTs automated indexing, as described in detail in the “Conclusion and Future Works” section.

4. Check Tags MeSH Terms (CTMTs) Indexing Project

4.1 Datasets

4.1.1 “Fully human indexed” citations

There are three MEDLINE indexing methods: “Automated”, “Curated”, and “Fully human indexed”. Both the “Automated” and “Curated” methods index articles algorithmically, but the “Curated” method includes an additional step where algorithm results are reviewed by a human. The “Fully human indexed” method, on the other hand, indexes articles solely by human indexers [1].

We used the 2021 MEDLINE Baseline dataset [12], which consists of about 30 million citations, to build the project datasets. To create a consistent and stable classifier for CTMTs, we used the “Fully human indexed”

citations to train and evaluate Deep Learning neural networks and ensemble machine learning models in this project. According to information posted in “Medical Subject Headings 2023" [13], the last Check Tags MeSH term “Young Adult” was included in MEDLINE in August 2008. Therefore, we only considered citations that were indexed after this date, and we collected 8,692,143 “Fully human indexed” citations for the project datasets.

4.1.2 Training, testing, and validating datasets

The 8,692,143 “Fully human indexed” citations are split into 85% for training (7,388,794), 10% for testing (868,982), and 5% for validating (434,367).

4.2 Feature vectors and their combinations of feature vectors

In this project, there are five different types of feature vectors that represent documents’ titles and abstracts. The first three feature vectors are open-source documents/sentences embeddings vectors, and the last two feature vectors are the project customized vectors.

1. 512-dimensional Universal Sentence Encoder (USE) feature vectors

   The Google Universal Sentence Encoder, based on the Transformer architecture [14], takes a concatenated document title and abstract and generates a 512-dimensional feature vector.

2. 384-dimensional Sentence Transformers Embeddings (STE) feature vectors

   The Sentence Embeddings using Siamese BERT-Networks [15] takes a concatenated document title and abstract and generates a 384-dimensional feature vector.

3. 700-dimensional Biomedical Sentence Embeddings (BSE) feature vectors

   The NLM BioSentVec pre-trained model [16,17,18] takes a concatenated document title and abstract and generates a 700-dimensional feature vector.

4. 25,600-dimensional Word-based Dictionaries (WBD) feature vectors

   Words in titles and abstracts of documents in the training dataset are collected and lemmatized using the spaCy NLP library software [19]. The lemmatized words are tabulated and ordered based on their occurrences from high to low. The top 25,600 lemmatized words are selected to build the Word-based dictionaries. A 25,600-dimensional WBD feature vector is then generated for each document by mapping lemmatized words and their occurrences in its title and abstract against those in the Word-based dictionaries.

5. 29,917-dimensional MeSH Entry Terms (ET) feature vectors

   Entry terms are “synonyms, near-synonyms alternate forms, and other closely related terms in a MeSH record” [20]. They are grouped together under the same MeSH term and generally used interchangeably with the preferred term in MeSH.

   For the 2021 MEDLINE Baseline dataset [12], there are 29,917 MeSH terms and each MeSH term is associated with a list of its entry terms. The 29,917-dimensional ET feature vector is generated for each document by searching for entry terms and their occurrences in its title and abstract.

To take advantage of the best performance of each feature vector on documents’ titles and abstracts, feature vectors other than the "baseline" feature vectors are not used alone but are combined into several groups of feature vectors combinations.

Based on experimental results, the WBD feature vectors performed better than other feature vectors in terms of precision and recall on the validating dataset. Therefore, the WBD feature vectors were chosen as the “baseline” feature vectors.

The “baseline” feature vectors are combined with the remaining four feature vectors to create 16 combinations of feature vectors, which are shown in Table 1.

4.3 CTMTs-based entry terms binary feature vectors

In addition to the 29,917-dimensional MeSH entry terms feature vectors defined in Section 4.2, a 40-dimensional CTMTs-based entry terms binary feature vector is generated for each document by searching for CTMTs entry terms information in its title and abstract.

4.4 Automated CTMTs Indexing Architecture

The automated CTMTs indexing architecture is designed as a two-level chained classifier based on Deep Learning neural networks and Random Forest Bagging machine learning.

The first level of the architecture takes a combination of feature vectors as input and predicts 40 CTMT outputs using Deep Learning neural networks. The second level of the architecture uses the predicted outputs of the first level, along with a CTMT-based entry terms binary feature vector as input and predicts each CTMT using Random Forest Bagging machine learning models.

The purpose of this design is to combine global and local classifications of CTMTs. The first level is considered as a global classification level, where feature vectors associated with all CTMTs are used together for their classifications. On the other hand, the second level is a local classification level, where feature vectors associated with each CTMT are used to classify it separately.

The automated CTMTs indexing architecture is shown in Figure 1.

4.4.1 Deep Learning Neural Networks Architecture and Configurations

The Deep Learning neural network architecture used in this project is a simple multi-layer fully connected neural network. The architecture has a combination of feature vectors as its input and 40 CTMTs as its outputs. Each feature vector input is either scaled up or down to a 512-dimensional vector, and then all 512-dimensional vectors are concatenated to input to the fully connected layers of the Deep Learning neural network.

As shown in Table 1, there are 16 deep learning neural networks, each trained on a different combination of feature vectors. The output dimension and total number of training parameters for each network are also listed in the table.

The Deep Learning models were implemented in Keras version 2.4.3. and the Deep Learning neural networks hyperparameters are listed in Table 2.

4.4.2 Training

The Deep Learning models were trained using the Adam optimizer and binary cross-entropy loss on a 16-core Intel(R) Xeon(R) CPU E5-2697 v4 @ 2.30GHz CPU with 125GB of memory.

Due to the large training dataset, the maximum number of epochs was set to 10, but the training would stop early if the F1-score did not improve. For this project, the average number of epochs for training a Deep Learning model was about 3.2, and the total training time for all Deep Learning models was about 60 hours.

4.4.3 "Spin-off" datasets

When human indexers performed MeSH indexing for MEDLINE, they used MeSH entry terms to help them determine appropriate MeSH terms for indexing articles. However, entry terms may not always appear in the title or abstract of a MEDLINE citation. Regarding the predictive performance of CTMTs, we found that citations with CTMTs-based entry terms in the title or abstract performed better than those without.

As a result, we decided to build the "spin-off" training, testing, and validating datasets by taking the project training, testing, and validating datasets and keeping only the records that have CTMTs-based entry terms found in their title and/or abstract.

Both datasets are trained by Deep Learning neural networks, and their predicted records are combined to create "combined" predicted records. As shown in Figure 4, the predictive performance of the "combined" predicted records is improved in terms of precision, recall, and F1-score.

4.4.4 Algorithm to combine Deep Learning predicted results from two datasets.

The project training dataset and its spin-off dataset are each trained by 16 Deep Learning neural networks to generate 32 sets of predicted records. These 32 sets are combined to build 16 combined-predicted records using the following algorithm:

Assume that there are X and Y records in the project dataset and its spin-off dataset, respectively.

```
For each of 16 Deep Learning neural networks:
....Train X records and build X predicted records for the project dataset.
....Train Y records and build Y predicted records for the project spin-off dataset.
....Set an array of XY-combined-predicted-record with a dimension X.
....Set an array index i to 0
....For each record in X predicted records:
........Get its X-PMID and its X-predicted-record.
........Set PMID_found_flag = False
........For each record in Y predicted records:
............Get its Y-PMID and its Y-predicted-record.
............If Y-PMID == X-PMID then
................XY-combined-predicted-record[i] = (X-predicted-record + Y-predicted-record) / 2 records
................Set PMID_found_flag = True
................break
........If PMID_found_flag == False then
............XY-combined-predicted-record[i] = X-predicted-record
........Increment i
...End For
End For
```

4.4.5 Random Forest Bagging Machine Learning Models Architecture

Random Forest Bagging machine learnings are ensemble methods that combine the predictions of multiple base estimators to improve performance. There are two types of ensemble methods: averaging methods and boosting methods. Averaging methods, such as "Bagging methods" and "Forests of randomized trees", build estimators independently and then take the average of their predictions. Boosting methods, such as "AdaBoost" and "Gradient Tree Boosting", train models sequentially with samples of randomized data, where each model uses the residuals of the previous model to minimize training errors.

The Random Forest Bagging machine learning model used in this project is the ensemble averaging method "RandomForestClassifier" [21]. It uses predicted outputs of Deep Learning neural networks and CTMTs-based entry terms to predict the output of each Check Tags MeSH term.

As shown in Fig. 3, there are 40 Random Forest Bagging machine learning blocks, one for each Check Tags MeSH term. Each block has a "RandomForestClassifier" random forest classifier that takes 16 predicted outputs from the 16 Deep Learning neural networks and a 40-dimensional binary array of CTMTs-based entry terms as input and predicts the output of the Check Tags MeSH term assigned to the block.

The hyperparameters "n_estimators" and "max_depth" of the "RandomForestClassifier" are set to 100 and 2, respectively.

5. Results

Figure 2 shows the precision, recall, and F1-score prediction performances of 16 Deep Learning neural networks on 40 Check Tags MeSH terms using the project testing dataset. The precisions and recalls reported

are very competitive among the 16 Deep Learning neural networks, but the F1-scores are improved when there are more combinations of feature vectors.

Figure 3 shows the precision, recall, and F1-score prediction performance of 16 Deep Learning neural networks on 40 Check Tags MeSH terms using the project "spin-off" testing dataset.

The results from Figures 2 and 3 are combined and shown in Figure 4. The thick lines represent results from the project testing dataset, and the thin lines represent results from the project "spin-off" testing dataset. As shown in Figure 4, the ranges of precision, recall, and F1-score of the project "spin-off" testing dataset are better than those of the project testing dataset. Therefore, the predictive performances are better for citations that have CTMTs-based entry terms in their title and/or abstract.

Figure 5 shows precision, recall, and F1-score prediction performance of 16 Deep Learning neural networks on 40 Check Tags MeSH terms using predicted records that are combined using the algorithm shown in Section 4.4.4. The prediction performance based on the project "combined" testing dataset is shown with thick red lines. In addition, the prediction performance based on the project testing dataset shown in Figure 2 is also shown with thin blue lines just for comparison purposes. In the "recall" area, the prediction performances of the project "combined" testing dataset and the project testing dataset are very competitive. However, in the precision and F1-score areas, the prediction performances of the project "combined" testing dataset are higher than those of the project testing dataset. Therefore, the "combined" predicted records using the algorithm shown in Section 4.4.4 help improve the prediction performances of 40 Check Tags MeSH terms.

Figure 6 shows the predictive performance of Random Forest Bagging Machine Learning classifiers for each CTMT and for all 40 CTMTs using the project testing dataset. The overall predictive performance of all 40 CTMTs is 86.0% precision, 81.1% recall, and 83.5% F1-score.

6. Discussion

This work presents an approach to automatically index MEDLINE Check Tag MeSH terms using two levels of classifications: a first level of Deep Learning neural networks classifiers and a second level of Random Forest Bagging machine learning classifiers. In both levels, the CTMTs-based entry terms data found in the title and/or abstract are used as inputs. The first level is trained on the project training dataset and its "spin-off" dataset (consisting of records that have CTMTs-based entry terms found in their title and/or abstract). The second level uses a 40-dimensional binary array of CTMTs-based entry terms as one of its inputs. Based on the experimental results, the CTMTs-based entry terms data used in these two levels improved the overall predictive performance of CTMTs by +0.48% in precision, -0.04% in recall, and +0.21% in F1-score on the project testing dataset. The increases in precision and F1-score as well as the decreases in recall are expected, as shown in Figure 5.

As shown in Figures 3 and 5, citations with CTMTs-based entry terms found in the title and/or abstract perform better than citations without. However, CTMTs-based entry terms information may not be present in the title and/or abstract for some MEDLINE citations, but it may be present in the full text. Therefore, to get closer to human indexers' performance on MeSH indexing, automated indexing methods must access the full text to collect relevant information (such as entry terms) for MeSH indexing.

There are 18 CTMTs that have the highest indexing occurrences among 40 CTMTs. They are ordered from high to low based on their indexing occurrences as follows: "Humans", "Female", "Male", "Animals", "Adult", "Middle Aged", "Aged", "Adolescent", "Young Adult", "Mice", "Child", "Aged, 80 and over", "Rats", "Child, Preschool", "United States", "Pregnancy", "Infant", "Infant, Newborn". Among these 18 CTMTs, the 10-age related CTMTs ("Adult", "Middle Aged", "Aged", "Adolescent", "Young Adult", "Child", "Aged, 80 and over", "Child, Preschool", "Infant", "Infant, Newborn) have the lowest predictive performance in average. Their average precision, recall, and F1-score are about 73.6%, 58.3%, and 64.2% respectively, as shown in Figure 6. To improve the overall predictive performance, the performance of these 18 CTMTs must be improved, especially the 10-age related CTMTs. One suggestion, which is mentioned in the next section "Conclusion and Future Works", is to classify each of 18 CTMTs separately and then combine their predicted records with the current 40 CTMTs binary outputs to improve the system performance.

Regarding the 10-age related CTMTs, the MeSH indexing instruction [22] required that when a range of ages appears in an article, it must be used to select the appropriate age group check tags. For example, "Infant" is from 1 to 23 months, "Infant, Newborn" is from 0 to 1 month, "Infant" is from 1 to 23 months, "Child" is from 6 to 12 years, "Child, Preschool" is from 2 to 5 years, "Young Adult" is from 19 to 24 years, "Adult" is from 19 to 44 years, "Middle Aged" is from 45 to 64 years, and so on. However, the range of ages information is often presented in the full text of an article, rather than in the title or abstract. Therefore, the availability of full text to automated indexing methods is an important factor in improving the overall predictive performance of the system.

7. Conclusion and Future Works

The automated indexing of 40 Check Tag MeSH terms using a two-level chained classifier - Deep Learning neural network multi-label classifiers followed by Random Forest Bagging machine learning classifiers - has been presented. In this design, two different machine learning classification approaches are chained to support global classifications and local classifications among 40 Check Tag MeSH terms. Feature vectors are combinations of open-source documents/sentences embeddings vectors and project customized vectors. They include document-based vectors, word-based vectors, and MeSH entry terms-based vectors.

As mentioned briefly in Section 3, this project provides a new perspective for handling multi-label classification problems by (1) combining embedded and customized input feature vectors, (2) merging datasets and their spin-off datasets to improve predictive performance, and (3) chaining two different machine learning classification approaches. As a result, this proposed architecture could establish a platform for possible future experiments to improve the overall performance of the CTMTs automated indexing. These experiments could include:

i. Experimenting with new embedding feature vectors and/or new customized feature vectors.
ii. Adding more records into the CTMTs-based entry terms spin-off datasets, if possible, for better predictive performance.
iii. Exploring different Deep Learning neural networks architectures, such as Convolutional Neural Network and/or different hidden neural networks layers configurations.
iv. Considering other Boosting/Bagging machine learning models, such as AdaBoost, Gradient Tree Boosting, other Bagging methods, and XGBoost, and experimenting with their combinations.
v. Building classifiers for each of the 18 CTMTs mentioned in the "Discussion" section separately, and then combining their predicted records with the current 40 CTMTs binary outputs for improvements.
vi. Building and classifying titles and abstracts sentences and combining their predicted records for improvements.

	Combination of Features Vectors Input Dimension	Output Dimension	Total Neural Network Training Parameters
1	25600	40	13,520,152
2	25600 & 512	40	14,701,080
3	25600 & 29917	40	30,019,096
4	25600 & 384	40	14,898,200
5	25600 & 700	40	15,059,992
6	25600 & 512 & 29917	40	31,592,984
7	25600 & 512 & 384	40	16,472,088
8	25600 & 512 & 700	40	16,633,880
9	25600 & 29917 & 384	40	31,790,104
10	25600 & 29917 & 700	40	31,951,896
11	25600 & 384 & 700	40	16,831,000
12	25600 & 512 & 29917 & 384	40	34,937,368
13	25600 & 512 & 29917 & 700	40	35,099,160
14	25600 & 512 & 384 & 700	40	19,978,264
15	25600 & 29917 & 384 & 700	40	35,296,280
16	25600 & 512 & 29917 & 384 & 700	40	40,541,208

Table 1. Combination of Feature Vectors, Output Dimension, and Training Parameters

Deep Learning Hyperparameter	Value
Loss function	binary_crossentropy
Activation for hidden layers	ReLU (Rectified Linear Activation)
Activation for output layers	Logistic (Sigmoid)
Learning rate	0.001

Table 2: Deep Learning Neural Networks Hyperparameters

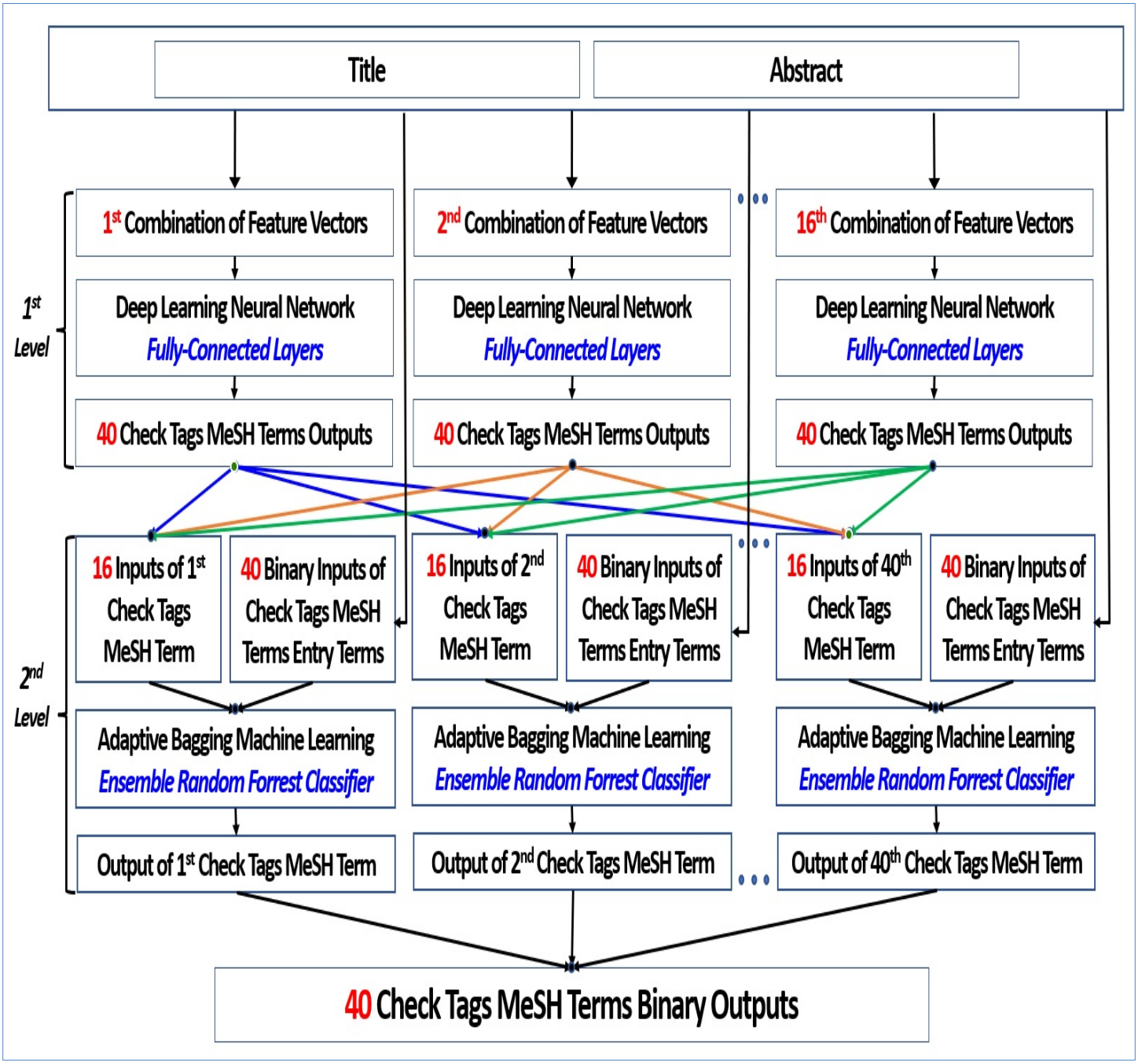

Figure 1: Automated Check Tags MeSH Terms Indexing Architecture

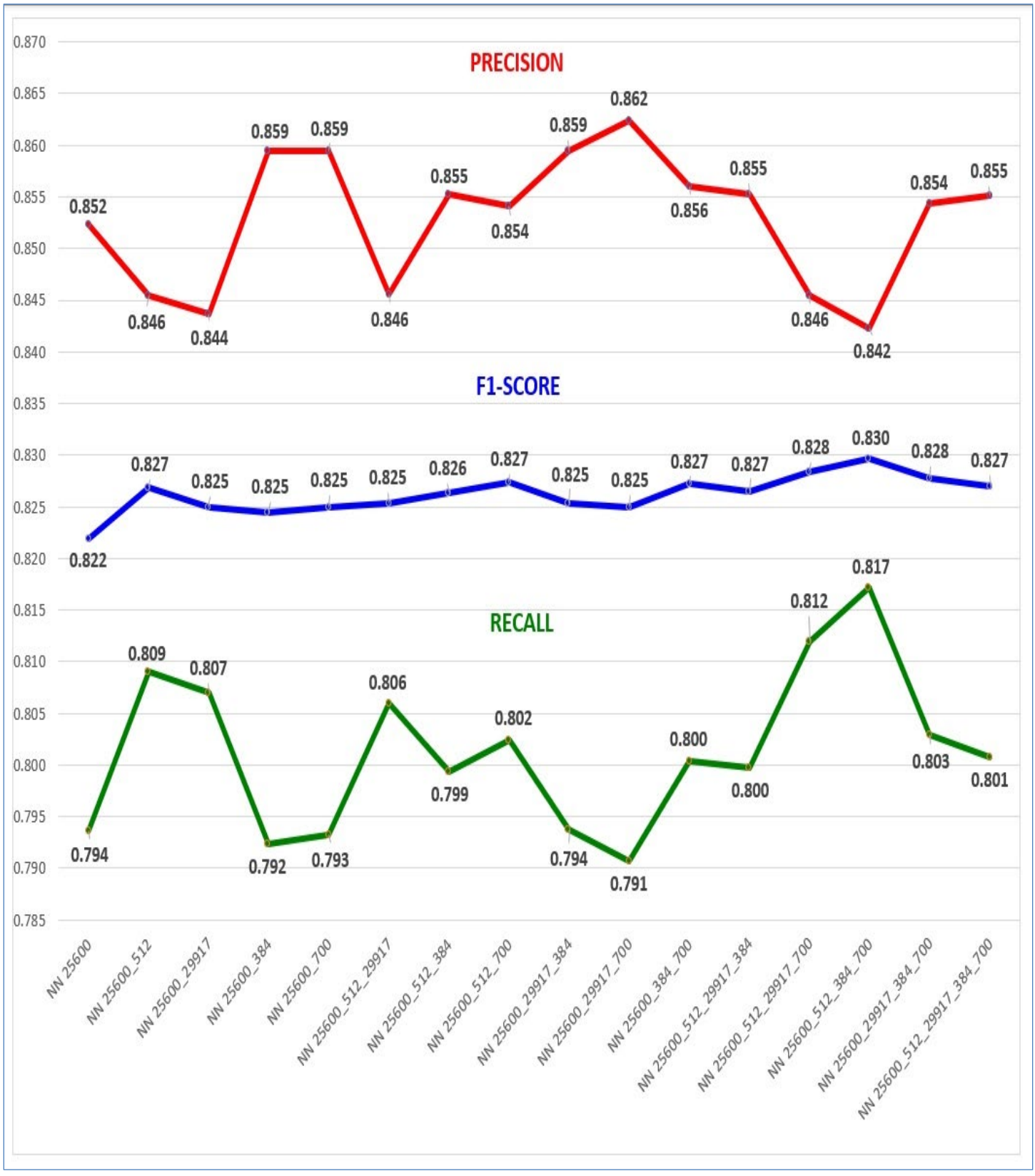

Figure 2: Performance of 16 Deep Learning Neural Networks on 40 CTMTs using the project testing dataset.

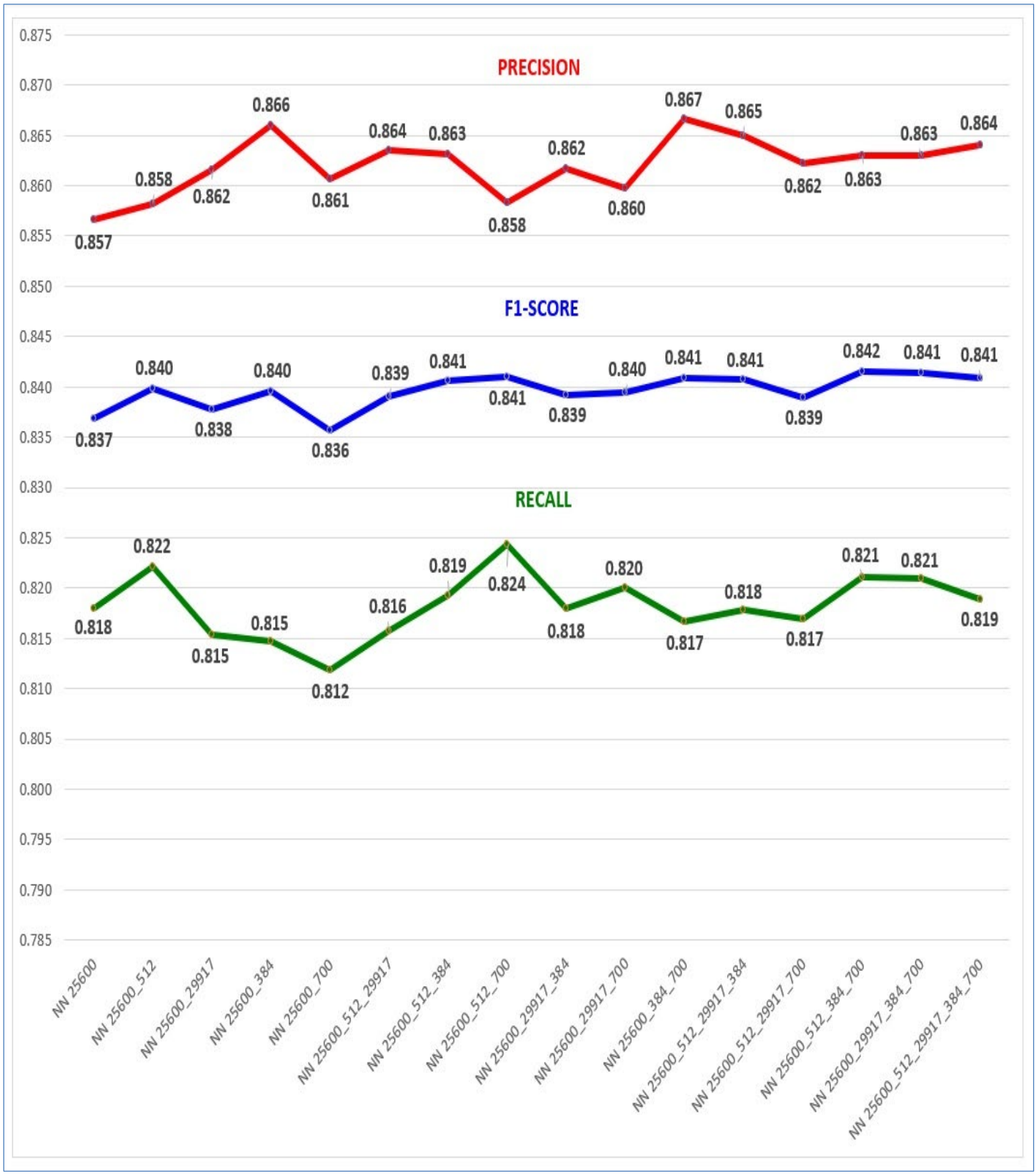

Figure 3: Performance of 16 Deep Learning Neural Networks on 40 CTMTs using the project "spin-off" testing dataset.

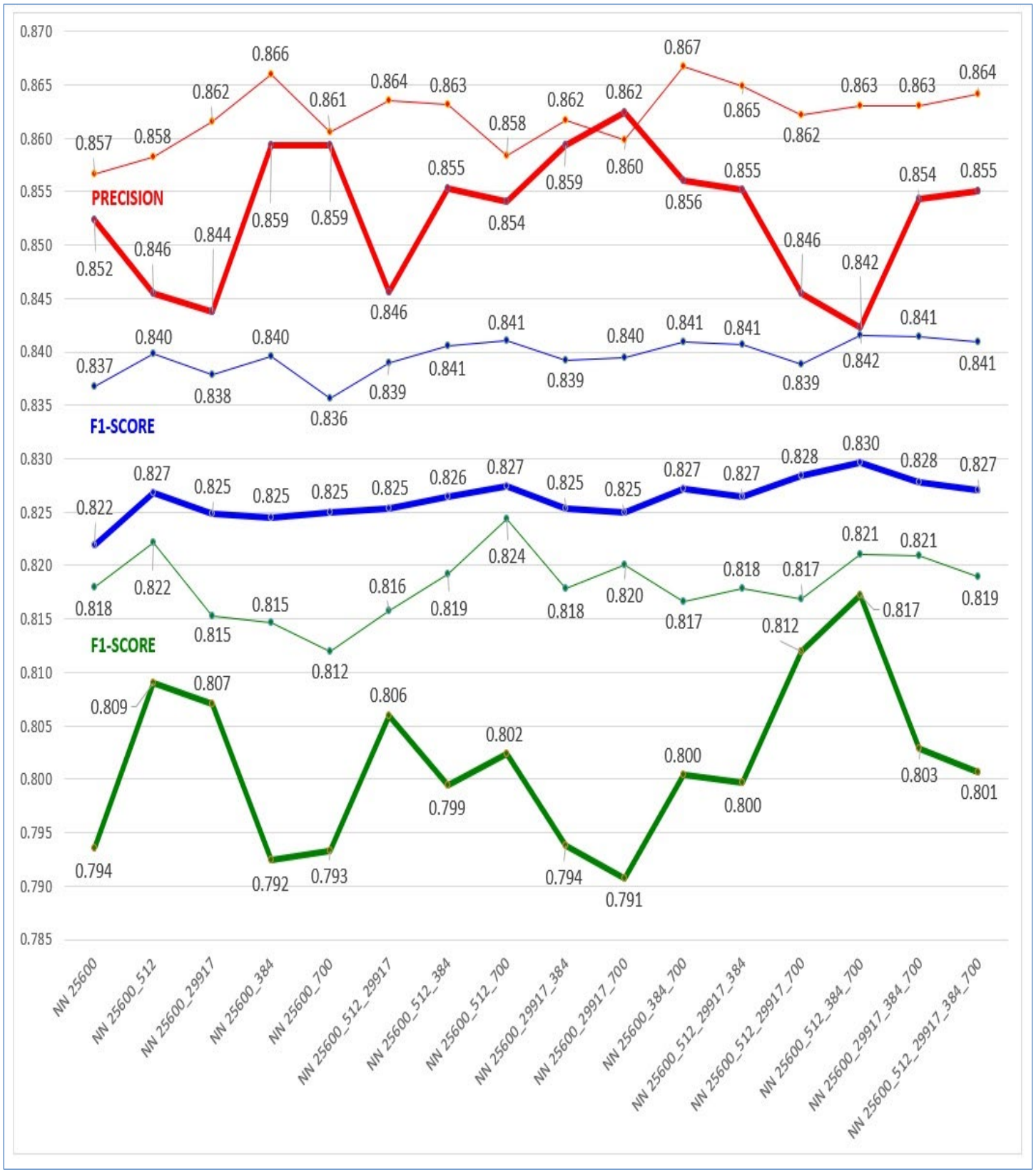

Figure 4: Performance of 16 Deep Learning Neural Networks on 40 CTMTs using both the project testing dataset and the project "spin-off" testing dataset.

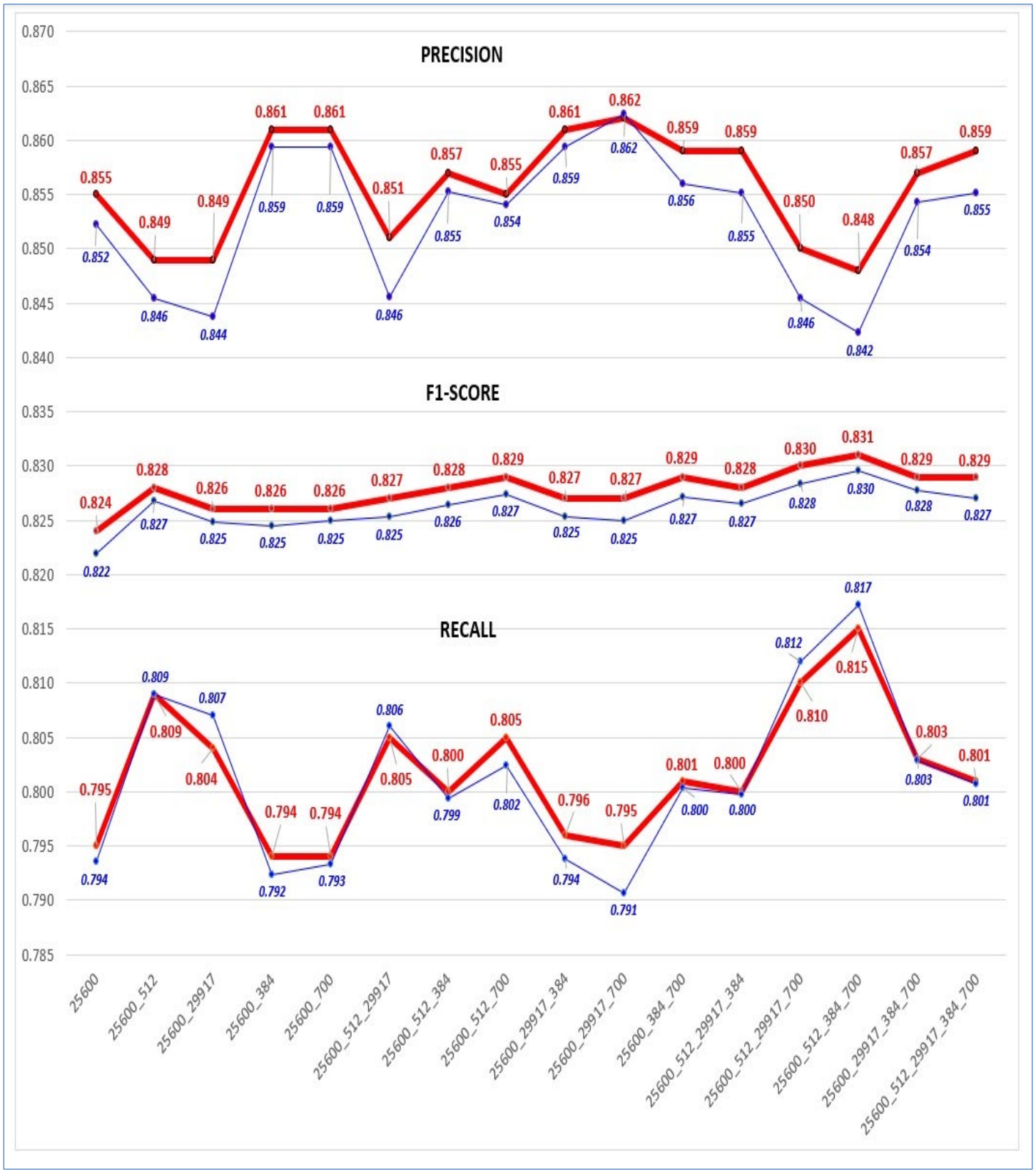

Figure 5: Performance of 16 Deep Learning neural networks on 40 Check Tags MeSH terms using predicted records that are combined by the algorithm shown in Section 4.4.4.

CTMTs	TP	TN	FN	FP	Precision	Recall	F1-Score
Adolescent	36,048	778,670	41,574	12,690	0.740	0.464	0.571
Adult	150,746	628,015	41,882	48,339	0.757	0.783	0.770
Aged	102,012	698,631	33,876	34,463	0.747	0.751	0.749
Aged, 80 and over	17,365	805,067	36,541	10,009	0.634	0.322	0.427
Animals	204,444	619,521	26,503	18,514	0.917	0.885	0.901
Bees	615	868,198	107	62	0.908	0.852	0.879
Cats	1,430	867,081	387	84	0.945	0.787	0.859
Cattle	6,321	858,857	2,763	1,041	0.859	0.696	0.769
Chlorocebus aethiops	444	866,639	1,757	142	0.758	0.202	0.319
Chick Embryo	507	867,854	479	142	0.781	0.514	0.620
Child	35,351	807,778	17,282	8,571	0.805	0.672	0.732
Child, Preschool	18,661	831,190	12,802	6,329	0.747	0.593	0.661
Dogs	4,770	862,668	1,202	342	0.933	0.799	0.861
Female	298,107	479,679	50,396	40,800	0.880	0.855	0.867
Guinea Pigs	809	867,813	315	45	0.947	0.720	0.818
Cricetinae	1,034	866,246	1,421	281	0.786	0.421	0.549
History of Medicine	-	868,926	56	-	0.000	0.000	0.000
Horses	1,531	866,985	390	76	0.953	0.797	0.868
Humans	583,877	234,891	26,621	23,593	0.961	0.956	0.959
Infant	13,297	840,085	10,768	4,832	0.733	0.553	0.630
Infant, Newborn	9,243	848,892	8,437	2,410	0.793	0.523	0.630
Male	294,738	480,265	47,078	46,901	0.863	0.862	0.862
Middle Aged	154,617	641,952	33,061	39,352	0.797	0.824	0.810
Pregnancy	19,541	842,317	4,475	2,649	0.881	0.814	0.846
Rabbits	3,369	864,166	1,174	273	0.925	0.742	0.823
Sheep	1,698	866,334	615	335	0.835	0.734	0.781
Swine	5,613	860,897	1,313	1,159	0.829	0.810	0.820
United States	11,014	842,869	11,431	3,668	0.750	0.491	0.593
History, 15th Century	-	868,805	177	-	0.000	0.000	0.000
History, 16th Century	-	868,707	275	-	0.000	0.000	0.000
History, 17th Century	-	868,608	374	-	0.000	0.000	0.000
History, 18th Century	-	868,453	529	-	0.000	0.000	0.000
History, 19th Century	661	867,167	848	306	0.684	0.438	0.534
History, 20th Century	1,832	864,708	1,714	728	0.716	0.517	0.600
History, 21st Century	-	867,087	1,895	-	0.000	0.000	0.000
History, Ancient	138	868,271	497	76	0.645	0.217	0.325
History, Medieval	-	868,682	300	-	0.000	0.000	0.000
Mice	57,385	788,251	12,638	10,708	0.843	0.820	0.831
Rats	32,465	824,890	6,031	5,596	0.853	0.843	0.848
Young Adult	25,617	777,893	48,737	16,735	0.605	0.345	0.439
OVERALL	**2,095,300**	**31,834,008**	**488,721**	**341,251**	**0.860**	**0.811**	**0.835**

Figure 6: Performance of Random Forest Bagging Machine Learning Classifiers for each CTMT and for 40 CTMTs using the project testing dataset.

References

1. https://www.nlm.nih.gov/bsd/indexfaq.html
2. Mork,J. et al. (2017) 12 years on–is the NLM medical text indexer still useful and relevant? J. Biomed. Seman., 8, 8.
3. Aronson,A. et al. (2004) The NLM indexing initiative's medical text indexer. Stud. Health Technol. Inform., 107, 268–272.
4. Tsoumakas,G. et al. (2013) Large-scale semantic indexing of biomedical publications. In: Proceedings of the First Workshop on Bio-Medical Semantic Indexing and Question Answering, a Post-Conference Workshop of Conference and Labs of the Evaluation Forum 2013, Valencia, Spain.
5. Tang, L. et al. (2009) Large scale multi-label classication via metalabeler. In: WWW '09: Proceedings of the 18th international conference on World wide web, New York, NY, USA, ACM 211-220
6. Mao,Y. et al. (2017) MeSH Now: automatic MeSH indexing at PubMed scale via learning to rank. J. Biomed. Seman., 8, 15.
7. Liu,K. et al. (2015) MeSHLabeler: improving the accuracy of large-scale MeSH indexing by integrating diverse evidence. Bioinformatics, 31, i339–i347.
8. Peng,S. et al. (2016) DeepMeSH: deep semantic representation for improving large-scale MeSH indexing. Bioinformatics, 32, i70–i79.
9. Jin,Q. et al. (2018) AttentionMeSH: simple, effective and interpretable automatic MeSH indexer. In: BioASQ@EMNLP. Brussels, Belgium, pp. 47–56.
10. Rae A. et al. (2019) Convolutional Neural Network for Automatic MeSH Indexing. PKDD/ECML Workshops (2): 581-594
11. Xun,G. et al. (2019) MeSHProbeNet: a self-attentive probe net for mesh indexing. Bioinformatics, 35, 3794.
12. Download MEDLINE/PubMed Data [internet] from www.nlm.nih.gov/databases/download/pubmed_medline.html
13. https://www.nlm.nih.gov/mesh/meshhome.html
14. https://tfhub.dev/google/universal-sentence-encoder-large/5
15. https://www.sbert.net/
16. https://github.com/ncbi-nlp/BioSentVec
17. Zhang Y. et al. (2019) BioWordVec, improving biomedical word embeddings with subword information and MeSH. Scientific Data.
18. Chen Q. et al. (2019) BioSentVec: creating sentence embeddings for biomedical texts. The 7th IEEE International Conference on Healthcare Informatics.
19. spaCy Linguistic features [internet]. 2021. Available from https://spacy.io/usage/linguistic-features
20. https://www.nlm.nih.gov/mesh/intro_entry.html
21. https://scikit-learn.org/stable/modules/ensemble.html#forest
22. MEDLINE Indexing Online Training Course [internet]. Available from https://www.nlm.nih.gov/bsd/indexing/training/CHK_030.html