Acta Scientific COMPUTER SCIENCES

Volume 3 Issue 10 October 2021

Research Article

# Applying Deep Learning for Hypotheses Generation to Bridge Medicinal Types Ayurveda and Allopathy using Deep Belief Network in Word Embedding

**Arockia Xavier Annie R* and Aishwarya T**

*Department of Computer Science and Engineering, College of Engineering, Anna University, India*

***Corresponding Author:** Arockia Xavier Annie R, Department of Computer Science and Engineering, College of Engineering, Anna University, India.

**Received:** August 04, 2021

**Published:** September 06, 2021



## Abstract

There is a humongous amount of information available in the biomedical field that leads to opportunities and several open issues. Biomedical research is towards the discovery of new cost efficient and side effects free drugs. Manual processing of this data could consume very high man-hours of labor. Automatic text processing of the abstracts of the research paper enhances the efficiency and could lead to the identification of interesting new hypotheses. Hence, hypotheses generation promises to shrink the number of possible substitutes that a medical researcher has to try in order to get a valid drug for a disease. Therefore, training word embedding using deep learning techniques improves the efficiency in vocabulary discovery and the classification of terms into various classes. The association between different diseases would serve as the base to derive hypotheses of drugs for the diseases. Our proposed work using Deep Belief Network (DBN) trained using the abstracts of the research papers has an accuracy of 82.98% whereas the traditional text processing method of hypotheses generation has an accuracy of 67.8%. Our work has provided symptom-disease, and disease-drug hypothesis that enables new alternatives and bridges between the Allopathy and Ayurveda drugs. It also intensifies and reduces the all possibilities in creation and testing of drugs for a particular disease. The hypothesis helps in reducing this all possibilities of drugs relevant to a disease.

**Keywords:** Disease-Drug Generation; Ayurveda and Allopathy; Deep Learning; Hypothesis Generation; Query Processing;

## Introduction

There is a large volume of biomedical information available on the internet. This has the potential to unlock new domains of knowledge in drug discovery as mentioned in [5]. If the drug in a philosophy has lurking side-effects, then the drug in another philosophy could be used to find a cure. Potential cheaper substitutes could be found for costlier drugs, if this knowledge is exploited fully was mentioned in [15,23,25]. The exact chemical compound that heals a disease can be found by comparing the chemical constitution of similar drugs. Hypotheses generation narrows down the number of possible drugs that could be tried instead of an existing one mentioned in [22]. If the drug A is used to treat the disease B and if disease B and disease C are related, then drug A could be used to treat disease C as well. This is the underlying key principle used in identifying new hypotheses noted in [12]. Automated Vocabulary Discovery (AVD) algorithm along with the computation of transitive closure yields new hypotheses. But the quality of the vocabulary discovered could be improved when it is combined with deep learning techniques. Training word embedding for deep learning has been used in finding out drug-drug interaction and in gene mentioned in [29,34].

Deep belief network (DBN) consisting of three layers is used for predicting drug-drug interaction [21,30]. The problem of finding disease-disease interaction is similar to it and could be solved using Deep Belief Network [16,18,19]. Hence, having a system consisting of two phases namely the hypotheses generation phase and the hypotheses querying phase is more precise, clear and user-friendly [4,32].

The first phase would involve the extraction of abstract from various research papers. Followed by this, these abstracts would be given as input to the traditional hypotheses generation and the Word Embedding hypotheses generation methods. The abstracts are tokenized, parsed to group biomedical terms together and subjected to the hypotheses generation methods respectively.

The latter phase examines the input query and converts complex queries to multiple simpler ones, handles spelling mistakes and then processes each of the simple queries and displays results according to both the methods of hypotheses generation as mentioned in [23].

There is a large volume of biomedical information available on the internet. This has the potential to unlock new domains of knowledge in drug discovery [7]. But, the manual extraction of information is not feasible. Chemical compounds are responsible for curing diseases irrespective of the domain of medicine types as Ayurveda or Allopathy [29]. The type of chemical compounds used, the way they are generated might vary depending on the type of medical philosophy. Hence the cost of a drug and its side effects may vary in each philosophy. Our system helps in comparing drugs given for the same disease in Allopathy and Ayurveda and thereby suggesting best result for research purpose.

Medical hypotheses generation helps the medical practitioners to experiment with administering new drugs to the patients. This aids in shifting from one form of medicine to another and to identify chemical compounds that actually heal the people. For example if fever is caused as an effect of cold, then the drugs that heal cold could be administered to heal fever. If the drug in a philosophy has lurking side effects, then the drug in another philosophy could be used. The exact chemical compound that heals a disease can be found by comparing the chemical constitution of similar drugs in two philosophies. The generated hypotheses would ease the research of drug discovery by limiting the number of drugs that could be used as possible substitutes for a drug with side effects or a costly drug is needed in [6].

To generate hypotheses of diseases and possible drugs in two systems of medicine namely, Allopathy and Ayurveda. This work also aims to analyze the capacity of using neural networks to work with biomedical word embedding and to evaluate the accuracy and the precision of the generated hypotheses.

The challenges faced in generating the hypotheses from the abstracts of biomedical research are as follows:

- The collection of a large dataset comprising of both Ayurveda and Allopathy medicines for processing is difficult.
- For the generation of hypotheses, it is difficult to come to a conclusion, only by analyzing a limited number of abstracts.
- These abstracts do not necessarily have the diseases and the drug names in scientific terms.
- Analyzing complex queries poses a challenge when the name of the disease is not a single word as, 'cold' and instead the name is like 'stomach ache'.
- Using the Standard English dictionary for query standardization is also not feasible and hence a separate medical dictionary has to be constructed.
- Manual intervention is required in sorting out certain discrepancies as there is no sufficient support for Ayurveda text processing.

Section 2 discusses the existing traditional approaches for hypotheses generation in detail. It also analyzes the advantages and disadvantages of each approach. Section 3 explains the system design with their algorithms and section 4 elaborates on its implementation setup and the results of the system and gives an idea of its efficiency. It also contains information about the data set used for testing and the observations made during testing. Section 5 concludes and gives an overview of its criticisms. Here, it states the various extensions that can be made to the system to make it function more effectively.

## Related work

Deep Belief Nets (DBN) has been increasingly used in a number of problems like image classification, speech recognition and audio classification. The DBNs are built by having Restricted Boltzmann Machine (RBM) as the building block as mentioned in [20]. DBN has an input layer, a number of hidden layers and the output layer. Each of these layers is an RBM stacked upon one another to form a DBN. Though harder to train, DBN has a higher modeling capacity than its shallow counterparts as mentioned in [20].

Deep learning methods are back into the limelight in the recent years. Deep and shallow learning methods could be differentiated by the number of layers used in building the neural networks as suggested in [9]. A neural network consists of a number of simple neurons each of which could be assigned a weight and a bias. Each of these neurons get activated by either the input or by the product of the weight and bias of the neurons in the previous layer, (i.e.) the output weight edge of the previous layer becomes the activating input for the next layer.

In [1,28] it is mentioned that large scale processing of biomedical literature promises to be a rich source of biomedical knowledge. Bag of Words (BoW) model is more effective in extracting information from the text. Named entity recognition combined with meta-data yields better results in computing genomic distance. The integration of available knowledge sources with complementary sources of information and computational methods for information extraction and processing helps in building better information retrieval systems in [32,34]. Automated Vocabulary Discovery (AVD) employs the Term-Frequency and Inverse-Document-Frequency (TF-IDF)method and categorizes the extracted words under various classes [31]. Transitive closure of the identified terms helps in deciding if there is a possibility for the generation of hypotheses as specified in [12,26,27]. Using the BFS algorithm, all the paths between any two terms could be computed. The existence of a path between two terms signifies the association or relationship between them and hence could be considered for the hypotheses generation.

Extraction of knowledge from biomedical literature involves finding novel relationship between objects. The problem of identification of interaction among the drugs could be considered as a multiclass classification problem as shown by [1,9,16,18,19]. DBN with a soft-max regression layer could be used to perform the classification of the terms in the respective classes. DBN is more efficient than Continuous Bag-of-Words (CBOW), Glove and Skip-gram models with a significant improvement in recall and hence, F-score [10], suggests that link analysis method could be used over the extracted features to gather new knowledge about the domain. The identification of links between two topics across documents forms the basis of hypotheses generation. Concept Association Graphs and Concept Chain Queries could be used to generate paths between any two concepts or terms.

For text mining systems a document representation is required which caters to the need of the particular application as mentioned by [31]. TF*IDF, Latent Semantic Indexing (LSI) and multiword are some of the text representations that are commonly used. The basic idea of TF*IDF is from the theory of language modeling that the terms in a given document can be divided into two categories: those words with Eliteness and those words without Eliteness as noted in [11]. Eliteness is evaluated using the TF and IDF values from [11]. TF*IDF has better statistical quality than LSI and multi-word representations in [31].

There is an increasing volume of unstructured data nowadays. The traditional database software fails in analyzing such large amounts of data as suggested by [13]. Hence, text mining methods are being developed to automate the process of analyzing data. The traditional systems of medicine such as Siddha and Ayurveda conceive human body as a holistic system as mentioned in [2]. Modern systems of treatment can benefit by incorporating the traditional knowledge from the other doctrines of medicine as mentioned in [14]. Medical overview of different philosophies when coexisting in an individual practitioner is more beneficial as stated in [12].

For almost many years before, the traditional use of natural products has led to the discovery of new drugs as mentioned by [33]. This poses a great opportunity to discover new cost-effective and side-effect free drugs. In India these other forms of medicines such as Ayurveda, Yoga, Naturopathy, Unani, Siddha and Homeopathy (AYUSH) system are also widespread and in use throughout different regions of the country as mentioned in [24]. But the knowledge of those is not fully exploited to the human benefit [8].

## System design

The system is partitioned into two phases, (i) Hypotheses generation phase and (ii) Hypotheses querying phase. Hypotheses generation phase consists of fetching diseases and drugs from Wikipedia.

As mentioned, the proposed system with ten modules is shown in Figure 1a in green, orange and blue blocks. The various modules in hypotheses generation phase are (i) fetch diseases and drugs from Wikipedia [35], (ii) extract the abstract, (iii) traditional hypotheses generation, (iv) tokenization, (v) extraction of stem, chunk and entity, (vi) training word embedding, (vii) generation of hypotheses bridging allopathy and Ayurveda. The traditional system hypotheses generation involves two submodules named as (iii) (a) Automated Vocabulary Discovery and (iii) (b) Tracing term associations as shown in Figure 1(b). Training word embedding includes the construction of a Deep Belief Network (DBN) and

training it which involves four steps as shown in figure 2. The DBN is composed of an (vi) (a) input layer, (vi) (b) three hidden layers and (vi) (c) logistic regression layer as shown in Figures 3. The second part of work is the hypotheses querying phase which has three modules namely (viii) query statement pre-processing, (ix) standardization and (x) query processing as shown in figure 1a.

**Figure 1a:** Proposed System Design.

**Figure 1b:** Traditional System hypothesis.

The flow of data across the hypotheses generation phase starts from the input keywords list of diseases, list of drugs and list of herbs given as query to the Wikipedia site [35]. The contents fetched are used as the input to extract relevant abstracts of research papers from the PubMed site [17]. These abstracts are then given as input to the traditional hypotheses generation to extract the hypotheses. The abstracts are tokenized, and later, parser is used to group stem, chunk and entities. Here we use Gdep parser. This is followed by training word embedding to calculate the score for the terms in the abstracts and group them into different classes. The scores discriminates the classes generated. The related terms are then analyzed to generate the other set of hypotheses.

In the hypotheses querying phase, the query given as input is preprocessed. If the given query is a complex one, it is converted to a set of simple queries and then processed. It is followed by query standardization, in which each of the terms is spell checked. Then the corresponding drugs from either of the hypotheses generation method would be retrieved depending on which the user demands.

User interface as shown in figure 3 is designed for easier access of the hypotheses and it consists of an input text box for giving the complex queries and two buttons for the user to drugs generated from the particular hypotheses would be displayed. Drugs generated from the particular hypotheses would be displayed, choose the method of hypotheses generation. On click of the button, the corresponding medications are displayed.

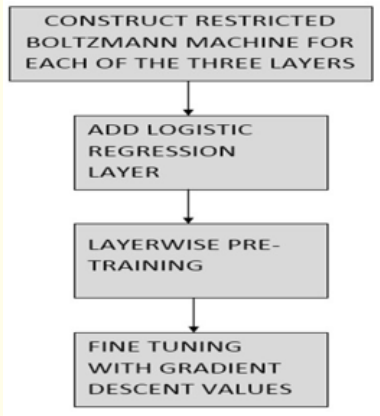


**Figure 2:** Training word embedding.

**Figure 3:** Deep Belief Network Layered Architecture.

## Modules split-up and detailed design

The details of implementing the proposed system are given as algorithms in each of the modules and its sub modules. These provide more information so as to make the working of our proposed system.

## Fetch diseases and drugs from wikipedia [35]

The list of diseases, drugs and medicinal plants has to be fetched from a public domain database. Wikipedia is considered as it is a reliable source containing all these details. Wikipedia python package provides a public API to query terms and to obtain corresponding results. Algorithm_1 uses the public API for Wikipedia and stores the results in a file for further proceedings.

- **Input:** List of drugs, List of diseases, List of medicinal plants
- **Output:** Dataset of disease names, drug names and herbs. E.g., Brain cancer, Lidocaine

Algorithm_1fetch-disease-drug(String keywords[])

1: list=null

2: file=open(newFilePath,w)

3: for i=1 to n do

4: keyword=keywords[i]

5: keyword=keywords[i]

6: end for

7: for each term in list do

8: file.write(term)

9: file.close()

10: end for

Fetch diseases and drugs from Wikipedia with list of drugs that was crawled from Wikipedia site [35]. Bisoprolol, Digoxin, Epinephrine, Lidocaine, Verapamil and Amiodarone are some of the medicines that are antiarrhythmic. Similarly many other medicines are fetched and stored in a file. Bile duct cancer, extrahepatic, bladder cancer, bone tumor, osteosarcoma are some of the examples of diseases.

## Extract the abstract from papers

The list of disease names, drugs and herbs are given as input to Algorithm_2 and the corresponding identifiers (id) of the related papers are obtained, which is then processed to get the abstract of those papers. PubMed [17], provides an open API to get the abstract of the research papers.

- **Input:** Dataset of disease names, drug names and herbs. E.g., cancer
- **Output:** Identifier (id) of related papers e.g., 28039958.

Abstract of all papers e.g., Current therapies for cancer.

Algorithm_2 extractAbstract()

1: open or create(data.txt)

2: get list of pid of the corresponding papers //pidpubmed id

3: for each pid do

4: url to download=http://www.ncbi.nlm.nih.gov/pubmed+pid

5: abstract=url to download.getAbstract()

6: write abstract to the text file

7: end for

Extract the abstract from papers from PubMed [17] using the crawled list of diseases and drugs. Author information and the abstract of the papers are written to a text file.

### Tokenization

Algorithm_3 is used for tokenizing the sentences in the abstract. Tokenizing involves removing spaces and special characters from the abstracts.

- **Input:** Abstract of all papers as text file. E.g., 26% of the patients
- **Output:** Tokens E.g., 26 of the patients

Algorithm_3 tokenization(abs.txt)

1: while !EOF(abs.txt) do

2: phrases by space = split using space(abstract)

3: phrases by char=split usingspecialcharacters(using space)

4: if phrases by char.contains(single char at end) then

5: t=strip end char(phrases by char)

6: tokens.add(t)

7: else

8: phrases by space.add(phrases by char)

9: end if

10: end while

### Tokenization

Tokenization involves the removal of symbols in the abstract and separating the sentences into words. For example in the sentence, 26% of cases can develop complications is separated as 26, of, cases, can, develop, complications.

### Extraction of stem, chunk and entity

Algorithm_4 uses the Genia dependency parser Gdep to group biomedical words together. In the traditional method this grouping of words is not included.

- **Input:** Tokens. E.g., major, nuclear, proteins
- **Output:** Stem, chunk and entity. E.g., major nuclear proteins (considered as a whole).

Algorithm_4 Extraction of stem, chunk and entity

1: Use GENIA Dependency parser for biomedical text

2: Get the input tokens from a file

3: Write the output stem, chunk and entity to the file

Extraction of stem, chunk and entity: Figure 3.1 shows the extraction of stem, chunk and entity using Genia Dependency parser. The output of the parser is used to group biomedical words together.

**Figure 4:** Extraction of stem, chunk and entity.

### Traditional hypotheses generation

Traditional hypotheses generation includes two sub-modules: (1). Automated Vocabulary Discovery (AVD) (2). Tracing term associations.

### Sub module 1: automated vocabulary discovery

Algorithm_5 is used to find the list of important words from the abstract. A list of unique words is found and then stop words are removed from that list. The tf-idf values are calculated and the important words are found.

- **Input:** Tokenized words
- **Output:** List of important words.

Algorithm_5 AVD(tokenized words)

1: file=open(new file path,w)

2: u words=unique(tokenized words)

3: remove stop words(u words)

4: file.write(w words)

5: file.close()

6: for each doc in file do

7: for each term do

8: find tf()

9: find idf()

10: calc tfidf()

11: end for

12: end for

13: return list of significant terms

Automated vocabulary discovery involves stop word removal and the calculation tf-idf calculation. Stop word removal is the process of removing words such as 'of', 'the', 'as', from the text. Figure 5 shows the output of tf-idf calculation. The words and their corresponding tf-idf values are displayed.

### Sub module 2: tracing term associations

Algorithm_6 computes the associations or the hypotheses existing in the abstracts. The transitive closure is computed to find the existence of an association and the paths are listed to derive at the hypotheses.

- **Input:** List of significant terms
- **Output:** Associations.

Algorithm_6 Tracing term associations (tokenized words)

1: for each term i in doc do

2: for each term j in doc do

3: compute transitive closure() 4: if

4: transitive closure(i,j) then

5: associations=list(path(i,j))

6: end if

7: end for

8: end for

9: return associations

The output of computing transitive closure, in the matrix that is displayed as output, the value of 1 indicates the presence of a transitive association and 0 indicates there is no connection between the two terms. Figure shows the paths that exist between the terms. For example, coriander could be used to treat malaria as there is a connection between malaria and diarrhea and diarrhea and coriander.

**Figure 5:** Tf-idf calculations.

### Training word embedding

- **Input:** Words
- **Output:** classified words E.g., disease: cancer, cold drug: Pepper, Aspirin cause: is caused by, responsible for negative: doctor, survivors, magnetic resonance imaging.

**Figure 6:** Listing paths.

The algorithm involves four sub-modules:

1. Construction of Restricted Boltzmann Machine (RBM) for each of the 3 layers
2. Addition of logistic regression layer
3. Layer wise pre-training
4. Fine tuning with gradient descent values

Figure 7 shows the screenshots while training the word embedding. As a result of training the similarity scores are obtained for each pair of the terms in the text as shown in figure 8.

### Sub module_1: Restricted Boltzmann Machine Construction

The input to algorithm_7 is the depth which is set to 3 and the output is the deep belief network consisting of three layers. The weight and bias values are initialized for the network.

**Figure 7:** Word embedding.

**Figure 8:** Score embedding.

- **Input**: Depth of layers
- **Output:** Deep Belief Network comprised of 3 Restricted Boltzmann Machine

Algorithm_7 Construction of rbm()

1: sigmoid layers=feed forward graph()

2: n layers=depth of module

3: feed forward graph=multi layerperceptron.hidden layers()

4: for i=1 to 3 do

5: rbm[i].weight matrix=sigmoid layers[i].weight_matrix

6: rbm[i].hidden bias=sigmoid layers[i].hidden bias

7: end for

### Sub module_2: Addition of logistic regression layer

Algorithm_8 adds a logistic regression layer on top of the three layered Deep Belief Network in order to convert it to a multi-class classification problem. Log likelihood values are computed in order to compute the fine tuning cost.

- **Input:** Sigmoid layers.output
- **Output:** Fully constructed Multi Layer Perceptron

Algorithm_8 Construction of logistic regression layer()

1: fine tune cost=log layer.negative log likelihood()

2: errors=log layer.errors()

3: sigmoid layers.add(logistic regression layer)

Sub module 3: Layer wise pre-training

Algorithm_9 is used to train the network. The network has to be trained layer by layer and correspondingly the timer is set to calculate the time taken for training.

- **Input:** Training data, batch size
- **Output:** Trained network.

Algorithm_9 Pretraining()

1: start time = timer.start – time()

2: for i=1 to 3 do

3: for each data in training_set do

4: pretrain fns()

5: end for

6: end for

7:end time = timer.endtime()

### Sub module 4: Fine tuning with gradient descent values

Fine tuning of the network is done in order to avoid the wrong values. Back propagation is time consuming and wastes a lot of computational resources. Hence fine tuning is done using gradient descent values as shown in algorithm 10.

- **Input:** Training set, Test set, valid set, learning rate
- **Output:** Modification in weight.

Algorithm_10 Fine tune with gradient descent values()

1: gradient=grad(fine tune cost, params)

2: update.append(param gradient*learning rate)

3: train(index, fine tune cost, update)

4: validate(index, errors)

5: test(index, errors)

### Generation of hypotheses

The hypotheses are generated using the algorithm_11. Every disease in the disease list is compared with other diseases and drugs to generate the hypotheses.

- **Input:** Classified diseases, drugs
- **Output:** Hypotheses stored in database.

Algorithm_11 Generation of Hypotheses()

1: for each d in disease list do

2: drug list=find associated drugs(d)

3: cause list=find associated diseases(d)

4: for each c in cause list do

5: drugs additional=find associated drugs (c)

6: end for

7: d.hypotheses of drugs=append(drug list, drugs additional)

8: end

Figure 9 shows the output of the hypotheses. The first column is the name of the disease and the second column indicates the name of the drug.

**Figure 9:** Generated hypotheses.

## Query statement preprocessing

The algorithm_12 is used to convert a complex query into a list of simple queries. A complex query could contain and/or. It is split to simple queries based on the presence of these.

- **Input:** Query E.g., fever, headache.
- **Output:** Pre-processed query E.g., fever, headache

Algorithm_12 Prepocessing(query)

1: remove punctuations(query)

2: if query.complex()==true then

3: query list=query.split to simple()

4: else

5: query list=query

6: end if

## Query standardization

Algorithm_13 is standardizes the queries. The simple queries are compared with the common medical terms and the correct term is replaced instead of the older term.

- **Input:** Pre-processed query E.g., couga
- **Output:** Standardized query E.g., cough.

Algorithm_13 Query standardization(preprocessed query)

1: construct BOW model of common medical terms

2: for each term pre processed query do

3: BOW model.search(term)

4: term.replace(correct terms)

5: end for

## Query processing

Query processing is done using algorithm 14.On giving the standardized query as the input, each of the simple query terms is processed and the list of drugs is fetched from the hypotheses generated using word-embedding and traditional methods and displayed as output to the user.

- **Input:** Standardized query
- **Output:** List of drugs.

Algorithm_14 Query processing(standardized query)

1: drug list = select drugs from disease=standardized query hypothesis disease drug where

2: while (drug list!=null) do

3:drug = drug list.head()

4: print drug

5: drug list=drug list.next()

6: end while

The user interface is used to get the input query from the user and display the list of drugs as the output. The buttons traditional or word embedding could be clicked by the user to view the drugs correspondingly generated from a particular method. The difference in the number of drugs found could be easily visualized with the help of the user interface.

The output of query standardization such as 'couga' is a misspell word that is corrected as cough using the dictionary. The output of query statement preprocessing in complex queries are converted to simple queries and then they are processed. For example cough and cold and cough, cold are some complex queries that are processed. The output of query processing, when disease name is given as input the hypotheses are retrieved. For example, when cancer is given as input then the corresponding drugs are displayed. The drugs from the hypotheses generated in both Traditional and Word embedding methods are displayed in figure 10.

**Figure 10:** Query processing.

## Implementation and Results

### Data set description

The dataset consists of abstracts from the PubMed [17]. These abstracts are collected by querying the PubMed database with a list of disease, drugs and herbs. The list of disease, drugs and herbs have been collected from Wikipedia [35]. The dataset consists of a summary of the abstract, the author name, the conclusion and the methods of the research papers which satisfy the query. As it requires a large amount of computational resources to consider the entire list, about 500 sample abstracts were considered.

### Experimental results

This chapter explains the output of various modules in hypotheses generation. Evaluation metrics such as accuracy, prediction, recall, Fscore, specificity and error rate are used to compare the efficiency of traditional and word embedding methods of hypotheses generation. The number of hypotheses generated for each of the diseases varies in both these methods. This difference is presented as a graph.

### Performance evaluation metrics

#### Accuracy

Accuracy refers to the closeness of a measured value to a standard or known value. Equation 4.1 gives the formula to calculate accuracy.

Accuracy = (TP+TN)/ (TP+TN+FP+FN)------(4.1)

Where TP, is True positive; FP, False Positive; FN, False Negative; TN, True Negative is the number of actual normal packets that were correctly classifies as normal.

#### Precision

Precision refers to the closeness of two or more measurements to each other. Precision can be measured using the formula mentioned in equation 4.2.

Precision (positive) = TP / (TP+FP)---------- (4.2)

#### Recall

Recall measures how many positive labels are successfully predicted amongst all positive labels. Recall can be calculated by substituting the values for true positive and false negative in the equation 4.3. This helps rule out disease (when the result is negative).

Recall =TP / (TP + FN)---------- (4.3)

#### F-score

It is a measure that combines precision and recall is the harmonic mean of precision and recall. F-score could be calculated using the formula as mentioned in the equation 4.8.

F = 2 * (Precision * Recall) / (Precision + Recall)----- (4.4)

#### Specificity

A very specific test rules in disease with a high degree of confidence. Specificity could be calculated by substituting the values in the formula given by equation 4.5.

Specificity =TN / (TN + FP)----------(4.5)

## Error rate

Error rate is a natural performance measure for classification problems. It is defined as the frequency of errors. Error rate could be calculated using either the formula given by equation 4.6 or 4.7 the equations are:

Error rate= 1−Accuracy----(4.6)

Error rate= (FP + FN) / (TN+TP+FP+FN)-----(4.7)

## Result analysis

Table 1 gives an overall view of the values that were obtained for traditional and word embedding methods of hypotheses generation. The numerical values clearly show that the word embedding method outperforms the traditional method of hypotheses generation.

| S. No. | Evaluation Metrics | Traditional Method | Word embedding method |
|---|---|---|---|
| 1 | Accuracy | 67.8 | 82.98 |
| 2 | Precision (positive) | 77.9 | 91.75 |
| 3 | Recall | 60.91 | 67.93 |
| 4 | F-score | 68.36 | 78.063 |
| 5 | Specificity | 69.38 | 84 |
| 6 | Error rate | 32.2 | 17.02 |
| 7 | Number of hypotheses generated | 181 | 315 |

**Table 1:** Evaluation metrics.

Figure 11a shows the accuracy of the two methods. The traditional method has an accuracy of 67.8% and the word embedding method has an accuracy of 82.98%. Figure 11b shows the precision of the two methods. The traditional method has a precision of 77.9% and the word embedding method has a precision of 91.75%. Figure 11c shows the recall of the two methods. The traditional method has a recall of 60.91% and the word embedding method has a recall of 67.93%. Figure 11d shows the f-score of the two methods. The traditional method has an f-score of 68.36% and the word embedding method has an f-score of 78.06%. figure 11e shows the specificity of the two methods. The traditional method has a specificity of 69.38% and the word embedding method has a specificity of 84%. Figure 11f shows the error rate of the two methods. The traditional method has an error rate of 32.2% and the word embedding method has an error rate of 82.98%.

Table 2 shows the disease names, the number of hypotheses generated in traditional method and word embedding method. About 39 diseases are listed and the corresponding numbers of hypotheses generated are listed. Figure 4.2 shows this table in graphical form. Here, red solid line indicates the disease-number of hypotheses generated using word embedding method. The blue dashed line indicates the disease-number of hypotheses generated using the traditional method. It is clearly visible that the word embedding method outperforms the traditional method in most of the cases.

| S. No. | Disease name | Number of Hypotheses Generated | |
|---|---|---|---|
| | | Traditional Method | Word Embedding Method |
| 1 | Cancer | 12 | 27 |
| 2 | Ache | 5 | 5 |
| 3 | Diarrhea | 4 | 4 |
| 4 | Cough | 3 | 3 |
| 5 | Arthritis | 1 | 2 |
| 6 | Diabetes | 6 | 15 |
| 7 | Joint pain | 1 | 4 |
| 8 | Wound | 1 | 1 |
| 9 | Tumor | 2 | 2 |
| 10 | Alzheimer | 13 | 19 |
| 11 | Hypercholesterolemia | 6 | 6 |
| 12 | Cold | 3 | 3 |
| 13 | Lipid peroxidation | 6 | 6 |
| 14 | Stress | 8 | 8 |
| 15 | Liver damage | 7 | 7 |
| 16 | Headache | 2 | 4 |
| 17 | Gastrointestinal | 5 | 5 |
| 18 | Stomach ache | 0 | 1 |
| 19 | Gout | 0 | 1 |
| 20 | Intestinal tumor | 0 | 3 |
| 21 | Lung cancer | 9 | 14 |
| 22 | Liver cancer | 13 | 14 |
| 23 | Breast cancer | 6 | 14 |
| 24 | Stomach cancer | 6 | 14 |
| 25 | Colorectal cancer | 2 | 14 |
| 26 | Cervix cancer | 8 | 14 |
| 27 | Prostate cancer | 9 | 14 |
| 28 | Hepatic fibrosis | 3 | 3 |
| 29 | Gastritis | 5 | 7 |
| 30 | Peptic ulcer | 4 | 8 |
| 31 | Gastric cancer | 4 | 8 |

| | | | |
|---|---|---|---|
| 32 | Parkinson | 4 | 9 |
| 33 | Multiple sclerosis | 4 | 9 |
| 34 | Brain tumor | 5 | 9 |
| 35 | Meningitis | 5 | 9 |
| 36 | Ileus | 0 | 3 |
| 37 | Cholesterol | 5 | 16 |
| 38 | Obesity | 2 | 5 |
| 39 | Art hero sclerosis | 2 | 5 |

**Table 2:** Disease Hypotheses.

Sometimes, the two lines coincide indicating that the number of hypotheses generated by both the methods is the same. This might be because of the fact that the number of times that particular disease is referred to in the chosen subset of abstracts might be comparatively lesser than the other diseases. In any case, the number of hypotheses found by word embedding is not lesser than that found by traditional hypotheses generation. The straight lines in the graph indicate that those diseases have equal number of hypotheses generated. This might be an entirely stochastic event or might be due to the fact that these diseases are referred to in the same number of similar abstracts.

**Figure 11**

**Figure 12:** Disease Hypotheses Chart.

## Conclusion

Hypotheses generation narrows down the number of possible drugs that could be tried and tested as substitutes for a given drug. It helps the medical practitioners to easily experiment with administering similar drugs to the patients. The chemical composition of the drugs that are derived as a result of hypotheses could be analyzed to test for similarities. Needless to say hypothesis generates more information that could be used in Drug Discoveries. Training of word embedding using the abstracts of the documents improves efficiency in identifying the related diseases and the drugs. A three layered deep belief network (DBN) for training word embedding helps in improved efficiency. Then transitive closure, BFS is computed to find the existence of new hypotheses, all possible hypotheses that could be present between the diseases and the drugs. If a disease is given as input then a list of drugs or herbs that could be used considered for the treatment of the disease is retrieved and displayed. The hypothesis that is generated using the traditional method and using our proposed method is done and the result from our proposed system is more efficient with 82.98% accuracy.

## Summary

- Generates hypotheses of the diseases and the possible drugs in two systems of medicine, namely Allopathy and Ayurveda
- Analysis on using neural networks to work with biomedical word embedding
- Evaluates the accuracy and the precision of the generated hypotheses in both the traditional and word embedding methods.

## Bibliography

1. Action L., *et al.* "Use of a machine learning framework to predict substance use disorder treatment success". *Plos One* 12.4 (2017): e0175383.

2. Ayyadurai VS. "The control systems engineering foundation of traditional Indian medicine: the Rosetta stone for Siddha and Ayurveda". *International Journal of System of Systems Engineering* 5.2 (2014): 125-149.

3. Bleecker LG. "Hypothesis-generating research and predictive medicine". *Genome Research* 23.7 (2013): 1051-1053.

4. Camacho DM., *et al.* "Next-generation machine learning for biological networks". *Cell* (2018).

5. Chandra S. "Ayurveda research, wellness and consumer rights". *Journal of Ayurveda and Integrative Medicine* 7.1 (2016): 6-10.

6. Costa FF. "Big data in biomedicine". *Drug Discovery Today* 19.4 (2014): 433-440.

7. Goetz LH and Stork NJ. "Personalized medicine: motivation, challenges, and progress". F*ertility and Sterility* 109.6 (2018): 952-963.

8. Hollinger A and Juridical I. "Knowledge discovery and data mining in biomedical informatics: The future Is in integrative, interactive machine learning solutions". In Interactive knowledge discovery and data mining in biomedical informatics. Springer, Berlin, Heidelberg (2014): 1-18.

9. Jiang Z., *et al.* "Training word embedding's for deep learning in biomedical text mining tasks". In Bioinformatics and Biomedicine (BIBM), 2015 IEEE International Conference on (2015): 625-628.

10. Jinn W., *et al.* "A text mining model for hypothesis generation". In Tools with Artificial Intelligence, 2007. ICTAI 2007. 19th IEEE International Conference on 2 (2007): 156-162.

11. Kang D and Park Y. "Based measurement of customer satisfaction in mobile service: Sentiment analysis and VIKOR approach". *Expert Systems with Applications* 41.4 (2014): 1041-1050.

12. Lanier WL., *et al.* "Empiricism and rationalism in medicine: can 2 competing philosophies coexist to improve the quality of medical care?" *In Mayo Clinic Proceedings* 88.10 (2013): 1042-1045.

13. Lee S., *et al.* "An empirical comparison of four text mining methods". *Journal of Computer Information Systems* 51.1 (2010): 1-10.

14. Niehaus KE., *et al.* "Machine learning for the Prediction of antibacterial susceptibility in Mycobacterium tuberculosis". In Biomedical and Health Informatics (BHI), 2014 IEEE-EMBS International Conference on (2014): 618-621.

15. Niehaus WN., *et al.* "The PM&R Journal implements a social media strategy to disseminate research and track alternative metrics in physical medicine and rehabilitation". *PM&R* 10.5 (2018): 538-543.

16. Popova M., *et al.* "Deep reinforcement learning for de novo drug design". *Science Advances* 4.7 (2018): eaap7885.

17. Ravi D., *et al.* "Deep learning for health informatics". *IEEE Journal of Biomedical and Health Informatics* 21.1 (2017): 4-21.

18. Rifaioglu AS., *et al.* "Recent applications of deep learning and machine intelligence on in silico drug discovery: methods, tools and databases". *Briefings in Bioinformatics* (2018).

19. Sarikaya R., *et al.* "Application of deep belief networks for natural language understanding". *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)* 22.4 (2014): 778-784.

20. Schmidhuber J. "Deep learning in neural networks: An overview". *Neural Networks* 61 (2015): 85-117.

21. Schumacher A., *et al.* "Changing R&D models in research-based pharmaceutical Companies". *Journal of Translational Medicine* 14.1 (2016): 105.

22. Shih WJ and Lin Y. "On study designs and hypotheses for clinical trials with predictive biomarkers". *Contemporary Clinical Trials* 62 (2017): 140-145.

23. Srivastava SR., *et al.* "Mainstreaming of Ayurveda, Yoga, Naturopathy, Unani, Siddha, and Homeopathy with the health care delivery system in India". *Journal of Traditional and Complementary Medicine* 5.2 (2015): 116-118.

24. Spangler S., *et al.* "Automated hypothesis generation based on mining scientific literature". In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (2014): 1877-1886.

25. Terranova N., *et al.* "Application of Machine Learning in Translational Medicine: Current Status and Future Opportunities". *AAPS Journal* 23 (2021): 74 (2021).

26. Valka HGG and Mukhopadhyay S. "Hypotheses Generation Pertaining to Ayurveda Using Automated Vocabulary Generation and Transitive Text Mining". In 2009 International Conference on Network-Based Information Systems (2009) 200-205.

27. Versos KM. "Drawing on millions of biomedical journal publications to do predictive biology". In Big Data and Smart Computing (Big Comp), 2015 International Conference on (2015): 251-253.

28. Yao V., *et al.* "Enabling Precision Medicine through Integrative Network Models". *Journal of Molecular Biology* 430 (2018): 2913-2923.

29. Yao J., *et al.* "Predicting clinically promising therapeutic hypotheses using tensor factorization". *BioRxiv* (2018): 272740.

30. Zhang W., *et al.* "A comparative study of TF* IDF, LSI and multi-words for text Classification". *Expert Systems with Applications* 38.3 (2011): 2758-2765.

31. Zhang L., *et al.* "From machine learning to deep learning: progress in machine Intelligence for rational drug discovery". *Drug Discovery Today* (2017).

32. Zhao XF., *et al.* "A novel drug discovery strategy inspired by traditional medicine philosophies". *Science* 347.6219 (2015): S38-S40.

33. Zitnik M., *et al.* "Machine learning for integrating data in biology and medicine: Principles, practice, and opportunities". *Information Fusion* 50 (2019): 71-91.

34. "Wikipedia, a free encyclopedia".

35. "Pubmed central".

**Volume 3 Issue 10 october 2021**