Chemical Named Entity Recognition

Dheeraj Gattupalli
201330167
dheeraj.gattupalli@students.iiit.ac.in

Allaparthi Sriteja
201302139
sriteja.allaparthi@research.iiit.ac.in

Akhil Kumar Singh Thakur
201302181
akhilkumarsingh.thakur@research.iiit.ac.in

***Abstract*—Specific information on newly discovered compounds is often difficult to find in chemical databases. For example, drug research requires the knowledge of new molecules for developing new drugs. Researchers may also want to search for potential lead compounds or determine the function of the compound. Obtaining previous knowledge on chemicals, such as biological properties or toxic effects, can help in many aspects of drug development processes. In this report we would discuss various algorithms implemented to extract chemical information and chemical-related entities such as drug names and source materials from text in several domains such as bioinformatics and nanoinformatics. We compared the performances of these algorithms with each other.**

***Keywords*—Chemical NER, Patent Text Mining,**

I. Introduction

Patents provide a huge and steadily growing source of publicly available information and knowledge. For example, up to 14% of all patent applications deal with chemical compounds and their use in novel pharmaceutical or agricultural products. To extract this domain specific knowledge we are aiming to develop and apply automated knowledge extraction processes. The quality of such extracted information relies on a correct named entity recognition (NER) of chemical compounds mentioned in patent texts.

This recognition process represents a particular challenge due to a variety of reasons: First, the peculiar linguistic features of patent texts pose a challenge for any attempt to automatically extract information: sentences are often very long and may exhibit a high syntactic complexity when compared to other text documents. Second, there is a great variety in which chemical entities are referred to in texts: For example, there are both trivial, half trivial and systematic names for chemical compounds and classes, as well as formulas, registration numbers and trade names for drugs. Chemical names can be extremely long and may contain variations of meaningful punctuation symbols and parentheses. Different chemistry name types can even be mixed within one chemical expression.

In this project we aimed at measuring the quality of recognizing mentions of chemical entities in patent texts using different algorithms.

II. Datasets

A. Orgsyn

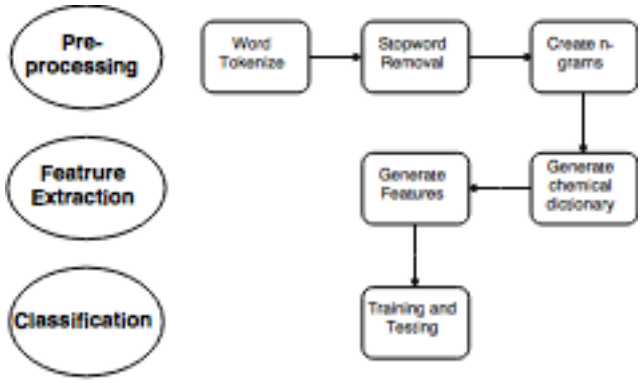
For training and testing of Classifier Models, Orgsyn dataset is used.Orgsy has a huge collection of chemical patents available in pdf format. For our project we collected the title and abstracts from these patents.

B. chem-train.csv

This file contains plain-text, UTF8-encoded Patent abstracts in a tab-separated format with the following three columns:
1. Patent identifier
2. Title of the patent
3. Abstract of the patent

C. chem-test.csv

This file contains plain text with tab seperated following two columns:
1. Patent identifier
2. Recognised Chemical Entity



III. Pre-processing details

Pre-Processing is essential for efficient Feature Extraction leading to non-redundant Feature Descriptors. The main steps involved in the Pre-Processing Stage of the Pipeline are :

- Word Tokenization
- Stopwords removal
- n-grams

A. Word Tokenization

Given a sentence a list of words are generated by considering space as delimiter. As many of the chemical entities consists of special characters, punctuation normal word tokenizers cannot give good results.

B. Stopwords Removal

Stop words refer to the most common words in a language. Although, there is no universal list for the Stop Words, but an exhaustive list has been used to ensure better accuracy and efficient feature extraction. They need to be removed because they are most likely to be not chemical names.

C. n-grams

The list of tokenized words is made into n-grams of size 1 to 6 by joining adjacent words together. As many of the chemical names consists of multiple words this step is necessary to extract these chemical entities.

IV. FEATURE EXTRACTION

After the pre-processing stage, sufficient amount of redundancy gets removed from the patent content. The words now need to be converted to equivalent feature descriptors to ensure good classification and low enough to ensure that computation performed on them is tractable.

Features:

A. Word and POS tags

The token itself (in lowercase) was added as a feature. POS tags were generated using nltk were also added as a feature.

B. Character Counts

Character counts of digits, uppercase and lowercase letters in each word.

C. Punctuation Characters

Presence of punctuation characters, dash, mathematical equals, mathematical identical.

D. Chemical Prefix and Suffix

Most frequently occurring prefixes, suffixes of length 3 and 4 were extracted from the systematic and trivial name dictionaries.
e.g. Prefixes: meth, etha, prop, etc. Suffixes: ane, ene, yne, etc.

E. Case Pattern Features

Feature created by replacing all uppercase letters with A, all lowercase letters with a and all digits with 0 upto length of 8 characters.

F. Character n-grams of Token

Character n-grams extracted from token of length 1 to 4 were added as features.
e.g. acetyl would have: a, ac, ace, acet, l, yl, tyl and etyl.

G. Presence in Dictionary

A binary feature for presence of Chemical element names, Chemical element symbols, Amino acid names, Amino acid codes of length 1 and 3, Systematic names, Trivial names, Family names, Greek letters and Greek symbols.

H. Molecular Formula Parser

A method to assign a numeric score to a string on scale of 0 to 1. Where a score of 1 indicates the string is 100% molecular formula and a score of 0 indicates this is not a molecular formula.
e.g. NaCl and PhNMe would have a score of 1.0000.

V. ALGORITHMS

A. Naive Bayes

In machine learning, naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes theorem with strong (naive) independence assumptions between the features.

A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness and diameter features.

Abstractly, naive Bayes is a conditional probability model: given a problem instance to be classified, represented by a vector x = (x_1, . . . , x_n) representing some n features (independent variables), it assigns to this instance probabilities

$$p(C_K|x_1, ..., x_n)$$

for each of K possible outcomes or classes. The problem with the above formulation is that if the number of features n is large or if a feature can take on a large number of values, then basing such a model on probability tables is infeasible. We therefore reformulate the model to make it more tractable. Using Bayes theorem, the conditional probability can be decomposed as

$$p(C_K|x) = \frac{p(C_K)p(x|C_K)}{p(x)}$$

$$p(C_K|x_1, ..., x_n) = \frac{1}{Z}p(C_k)\prod_{i=1}^{n} p(x_i|C_K)$$

where the evidence,

$$Z = p(x)$$

is a scaling factor dependent only on x_1, . . . , x_n, that is, a constant if the values of the feature variables are known. The discussion so far has derived the independent feature model, that is, the naive Bayes probability model. The naive Bayes classifier combines this model with a decision rule. One common rule is to pick the hypothesis that is most probable; this is known as the maximum a posteriori or MAP decision rule. The corresponding classifier, a Bayes classifier, is the function that assigns a class label y = Ck for some k as follows:

$$\hat{y} = \arg \max_{k \in (1,2,...,k)} p(C_k)\prod_{i=1}^{n} p(x_i|C_K)$$

B. Artificial Neural Network

In machine learning and cognitive science, artificial neural networks (ANNs) are a family of models inspired by biological neural networks (the central nervous systems of animals, in particular the brain) and are used to estimate or approximate functions that can depend on a large number of inputs and are generally unknown. Artificial neural networks are generally presented as systems of interconnected neurons which exchange messages between each other. The connections have numeric weights that can be tuned based on experience, making neural nets adaptive to inputs and capable of learning. We are given a set of example pairs $(x, y), x \in X, y \in Y$ and the aim is to find a function $f : X \to Y$ in the allowed class of functions that matches the examples. In other words, we wish to infer the mapping implied by the data; the cost function is related to the mismatch between our mapping and the data and it implicitly contains prior knowledge about the problem domain.

A commonly used cost is the mean-squared error, which tries to minimize the average squared error between the networks output, $f(x)$, and the target value y over all the example pairs. When one tries to minimize this cost using gradient descent for the class of neural networks called multilayer perceptrons, one obtains the common and well-known backpropagation algorithm for training neural networks.

Tasks that fall within the paradigm of supervised learning are pattern recognition (also known as classification) and regression (also known as function approximation). The supervised learning paradigm is also applicable to sequential data (e.g., for speech and gesture recognition). This can be thought of as learning with a teacher, in the form of a function that provides continuous feedback on the quality of solutions obtained thus far.

C. SVM

In machine learning, support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. Given a set of training examples, each marked for belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.
*3mm Writing the classification rule in its unconstrained dual form reveals that the maximum-margin hyperplane and therefore the classification task is only a function of the support vectors, the subset of the training data that lie on the margin. Maximize (in α_i)

$$L = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

D. Conditional Random Fields

Conditional random fields (CRFs) are a class of statistical modeling method often applied in pattern recognition and machine learning and used for structured prediction. CRFs fall into the sequence modeling family. Whereas a discrete classifier predicts a label for a single sample without considering "neighboring" samples, a CRF can take context into account; e.g., the linear chain CRF (which is popular in natural language processing) predicts sequences of labels for sequences of input samples.

It is often used for labeling or parsing of sequential data, such as natural language processing or biological sequences and in computer vision. Specifically, CRFs find applications in POS Tagging, shallow parsing, named entity recognition, gene finding and peptide critical functional region finding,among other tasks.

E. Random Forests

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

In particular, trees that are grown very deep tend to learn highly irregular patterns: they overfit their training sets, i.e. have low bias, but very high variance. Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance. This comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance in the final model.

VI. EVALUATION MEASURES

A. Precision

Precision (P) is the number of True Chemical Entities identified as a percentage of all Chemical Entities identified.

$$P = \frac{NumberOfChemicalEntitiesCorrectlyIdentified}{TotalNumberOfChemicalEntitiesIdentified}$$

B. Recall

Recall (R) is the percentage of all Chemical Entities that are correctly classified as Chemical Entities.

$$R = \frac{Numberofchemicalentitiescorrectlyidentified}{TotalNumberofChemicalEntities}$$

C. F1-Score

The F1 score (also F-score or F-measure) is a measure of a tests accuracy. It considers both the precision p and the recall r of the test to compute the score: p is the number of correct positive results divided by the number of all positive results, and r is the number of correct positive results divided by the number of positive results that should have been returned. The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst at 0.

$$F1 = 2 * \frac{precision * recall}{precision + recall}$$

VII. RESULTS

A. Naive Bayes

Parametrized By : Model to represent Data

	Recall	Precision	F-score
Non-chemical entities	0.99	0.95	0.97
chemical entities	0.68	0.78	0.72

B. Artificial Neural Network

Parametrized By : No. of Hidden Units

	Recall	Precision	F-score
Non-chemical entities	0.98	0.99	0.99
chemical entities	0.69	0.80	0.74

C. SVM

Parametrized By : Choice of Kernel

	Recall	Precision	F-score
Non-chemical entities	0.99	0.99	0.99
chemical entities	0.75	0.85	0.80

D. Conditional Random Fields

	Recall	Precision	F-score
Non-chemical entities	0.99	0.98	0.99
chemical entities	0.74	0.87	0.80

E. Random Forests

Parametrized By : Structure of Tree

	Recall	Precision	F-score
Non-chemical entities	0.99	0.98	0.99
chemical entities	0.72	0.80	0.78

REFERENCES

[1] https://taku910.github.io/crfpp/ *CRF++: Yet Another CRF toolkit*

[2] http://scikit-learn.org/stable/

[3] http://tartarus.org/martin/PorterStemmer/matlab.txt