Extraction of information using state of the art techniques in the context of various I2I POCs.

Abhibha Gupta
Dr. Sagar Sunkle
October 31, 2020

Index

Introduction

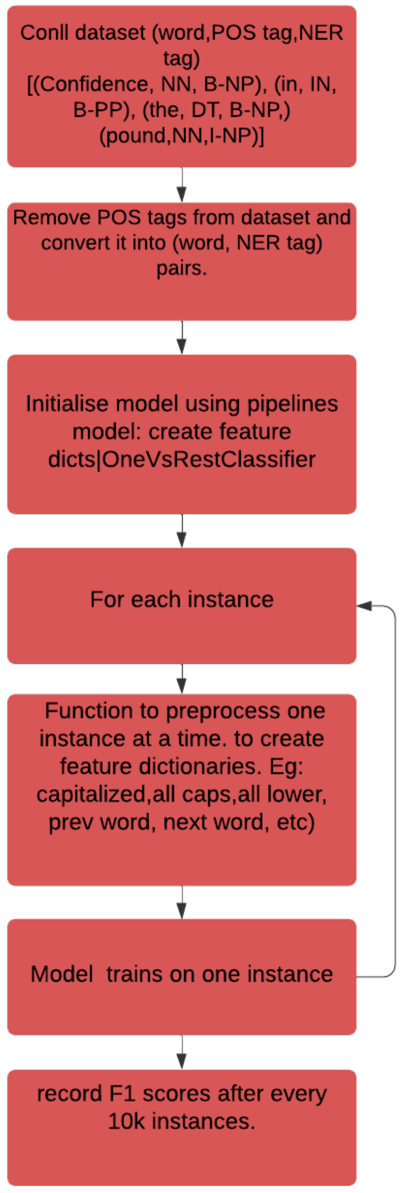
One of the main challenges of Natural language processing (NLP) is converting unstructured data into a structured format. Structured data can then in turn be used to create knowledge graphs, train other machine learning models, etc. A widely used method for this is **Named Entity Recognition (NER)**[1]. It involves the identification and extraction of some particular entities of interest in the text. Many trained models available are able to extract names of people, places and organization with great accuracy.
In our case we had a corpus of **600 research texts** regarding the different types of coatings applied on Steel. Our task was to populate a **domain model** consisting of predefined entities like ingredients used, there quantities, the conditions under which the steel coating process took place, coating type, substrate, etc. We started by creating our own corpus to train a NER model that could identify some of the basic entities like occurrences of molecules, processes, conditions, actions and quantities. Using the trained model we implemented an **Annotation tool cum Search tool** where one could perform queries, that would perform search on the database and return focused results. This enabled us to populate our domain model without training an NER for all entities. The workflow has been described in the subsequent sections.

[1] https://en.wikipedia.org/wiki/Named-entity_recognition

Exploration Work

POS tagging

Our main aim was to perform NER on the steel coatings text to extract entities to populate our domain model. To perform NER on our coatings text we were required to train our own custom model that catered to our need. Since the task of POS tagging is close to NER, we started out by using different POS tagging datasets like the **Conll dataset**[2], the **Penn dataset**[3] and the **POS tagging dataset from Kaggle**[4]. We started out by exploring an open source library called **Creme**.


Conll dataset (word,POS tag,NER tag) [(Confidence, NN, B-NP), (in, IN, B-PP), (the, DT, B-NP,) (pound,NN,I-NP)]
Remove POS tags from dataset and convert it into (word, NER tag) pairs.
Initialise model using pipelines model: create feature dicts|OneVsRestClassifier
For each instance
Function to preprocess one instance at a time. to create feature dictionaries. Eg: capitalized,all caps,all lower, prev word, next word, etc)
Model trains on one instance
record F1 scores after every 10k instances.

Creme: [5] is an open source library for online machine learning. Since it trains from an instance at a time, it is used to learn realtime from streaming data. We used different algorithms like AdaBoost[6], Naive Bayes[7], Random Forest[8], Decision Tree[9], The One Vs Rest Classifier[10], Ensemble Bagging Classifier[11] to experiment with the data. Out of these the One vs Rest Classifier (Classifier: Decision Tree classifier) gave the best result of a F1 score of 0.66 on the Conll dataset. We used

2 https://www.clips.uantwerpen.be/conll2003/ner/

3

4 https://www.kaggle.com/tarunpaparaju/jigsaw-competition-part-of-speech-tagging

5 https://github.com/creme-ml/creme

6 https://creme-ml.github.io/api-reference/ensemble/AdaBoostClassifier/

7 https://creme-ml.github.io/api-reference/naive-bayes/GaussianNB/

8 https://creme-ml.github.io/api-reference/tree/RandomForestClassifier/

9 https://creme-ml.github.io/api-reference/tree/DecisionTreeClassifier/

10 https://creme-ml.github.io/api-reference/multiclass/OneVsRestClassifier/

11 https://creme-ml.github.io/api-reference/ensemble/BaggingClassifier/

the pipeline approach to train our classifiers.

Pipeline Approach[12]: The creators of creme highly recommend to implement pipelines while training one’s model. In this method every instance before being used by the model to train passes through some steps of preprocessing (in our case creation of feature arrays) before being fed into the machine learning algorithm.

Since we weren’t getting satisfactory F1 scores we switched to other methods.

[12] https://creme-ml.github.io/user-guide/pipelines/

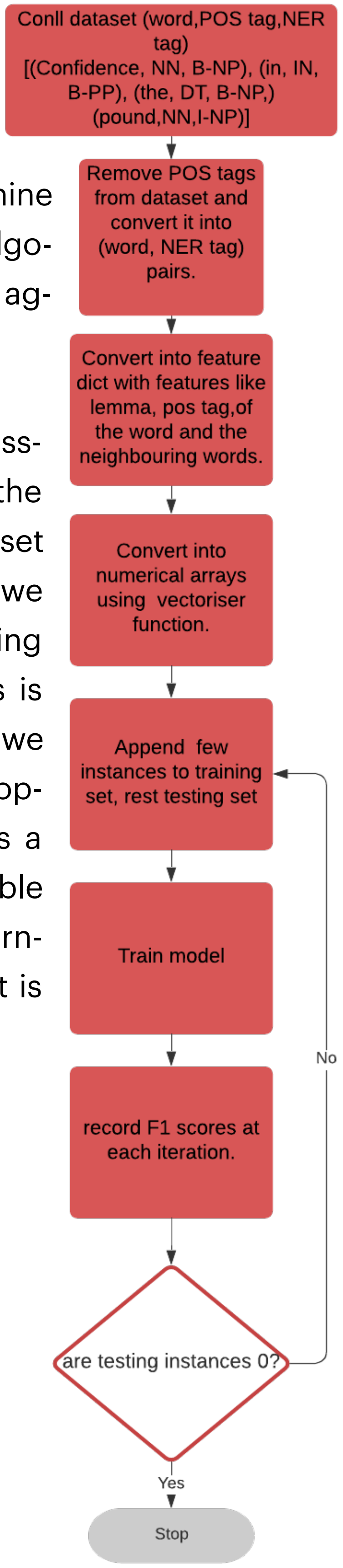
Exploration work

NER tagging

We now started with the task of NER on the **Conll dataset** [13] by exploring **Scikit learn.**

Scikit learn: It provides limited support for incremental machine learning. We used the partial fit method[14] on the Perceptron[15] algorithm to train the model in an incremental manner and got an aggregate F1 score of **0.96.**

Incremental machine learning: In this method after preprocessing and splitting the dataset into train-test sets we perform the training process by first taking a few instances in the training set and the rest in the testing set. For every iteration while training, we append a few instances to the training set and reduce the testing instances. We observe the F1 scores after every iteration. This is especially useful in cases where we have unannotated data and we want to find out how much data we need to annotate to get an optimal score. Usually after some iterations the F1 score reaches a threshold value and it does not increase by a considerable amount. At that point we can conclude that the model isn't learning anything new. If the trend of data for the rest of the dataset is the same then we don't need to annotate any more data.


Conll dataset (word,POS tag,NER tag) [(Confidence, NN, B-NP), (in, IN, B-PP), (the, DT, B-NP,) (pound,NN,I-NP)]
Remove POS tags from dataset and convert it into (word, NER tag) pairs.
Convert into feature dict with features like lemma, pos tag,of the word and the neighbouring words.
Convert into numerical arrays using vectoriser function.
Append few instances to training set, rest testing set
Train model
record F1 scores at each iteration.
are testing instances 0?
No
Yes
Stop

13 https://www.clips.uantwerpen.be/conll2003/ner/

14 https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Perceptron.html#sklearn.linear_model.Perceptron.partial_fit

15 https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Perceptron.html#sklearn-linear-model-perceptron

Preparation of Dataset

MW and AMCPE Structures

To train our own custom NER model we started by preparing a dataset. For now our basic target entities were molecules, quantities, actions, conditions and processes. We identified two particular relations of interest,
MW structures: Molecule-Weight/Quantity Structures and
AMCPE structures: Action-Molecule-Condition-Process-Extra Structures

The following example shows the output generated from each of the libraries **NLTK POS tagger**[16] and **Chemical Tagger**[17]

Sentence: To generate a Ni–P/Ni–B duplex coating, the surface of the HVOF-sprayed 420 martensitic stainless steel coating was prepared for plating by mechanical grinding, acetone degreasing and etching in a 30 vol.%HCl solution for 1 min.

MW - Structures:
Chemical Tagger
Output:<MOLECULE><OSCARCM><OSCAR-CM>**Ni-P**</OSCAR-CM><DASH>**/**</DASH><OSCAR-CM>**Ni-B**</OSCAR-CM></OSCARCM></MOLECULE>
<MOLECULE><OSCARCM><OSCAR-CM>**acetone**</OSCAR-CM></OSCARCM></MOLECULE>
<MOLECULE><QUANTITY><PERCENT><CD>**30**</CD><NN>**vol.**</NN><NN-PERCENT>**%**</NN-PERCENT></PERCENT></QUANTITY><OSCARCM><OSCAR-CM>**HCl**</OSCAR-CM></OSCARCM></MOLECULE>
<TimePhrase><IN-FOR>**for**</IN-FOR><CD>**1**</CD><NN-TIME>**min**</NN-TIME></TimePhrase>

[16] http://www.nltk.org/book/ch05.html

[17] http://chemicaltagger.ch.cam.ac.uk/

Obtained entities: **Molecules:** Ni-P/Ni-B, acetone, HCl

Quantity: - 30 vol.%

AMCPE Structures:

Chemical Tagger: Ni-P/Ni-B, acetone, HCl, for 1 min

NLTK POS Tagger: plating

Obtained entities:

Molecules: Ni-P/Ni-B, acetone, HCl

Actions: plating

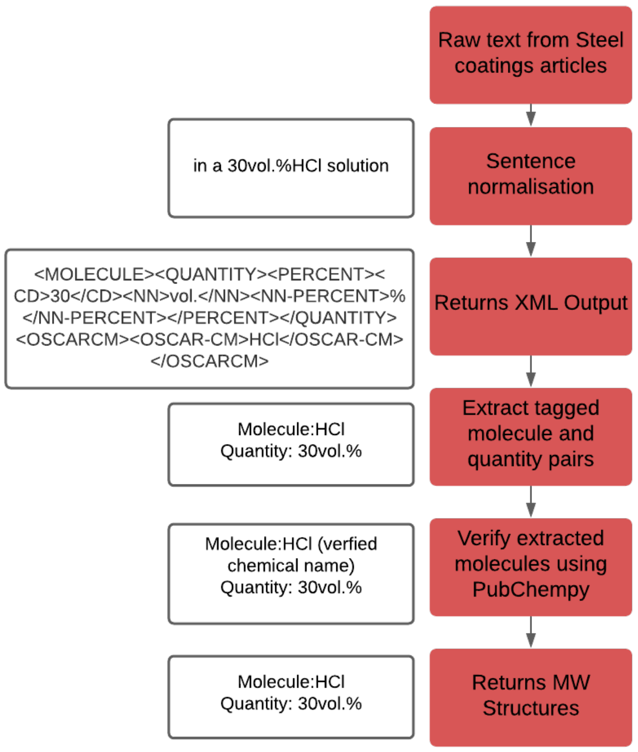
Conditions: for 1 min

Quantity: 30 vol.%

Process: -

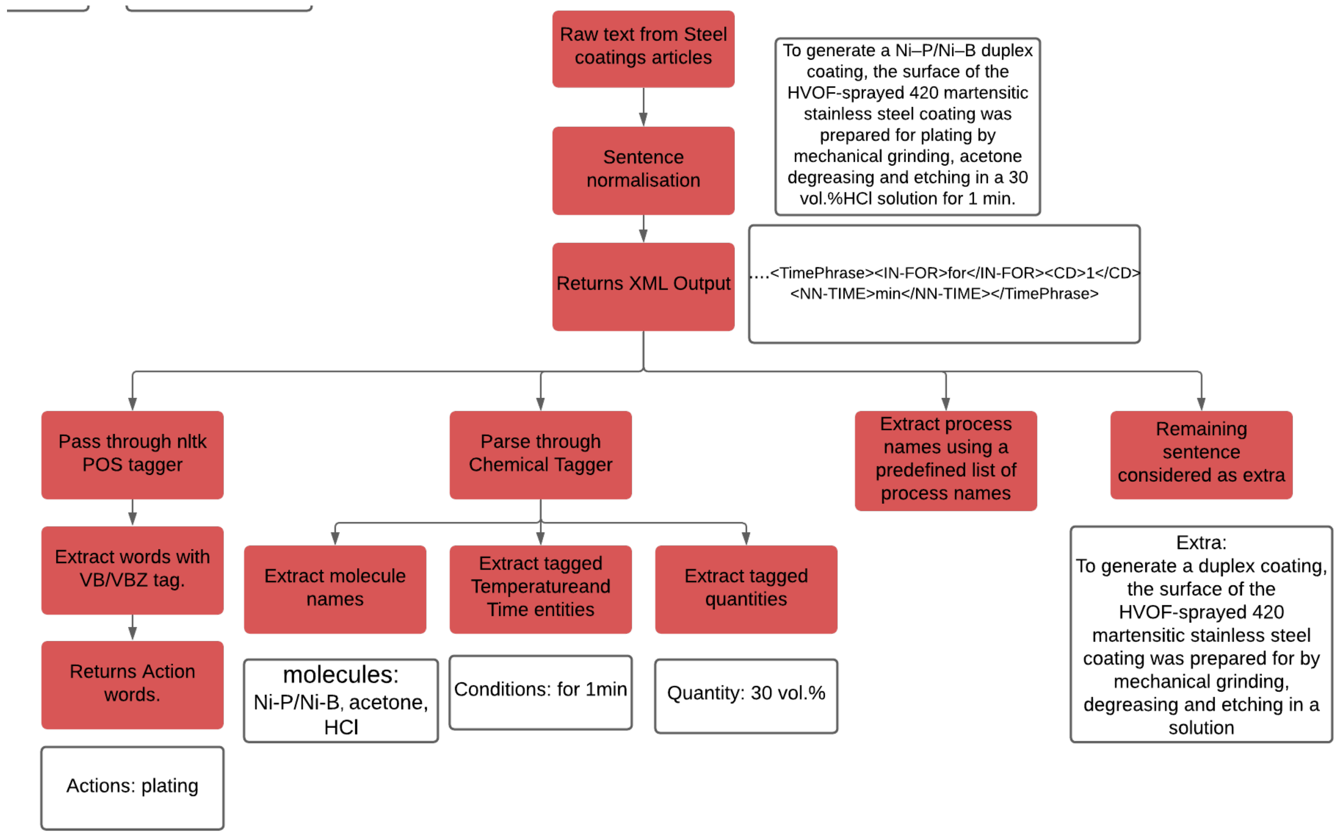
Extra: To generate a duplex coating, the surface of the HVOF-sprayed 420 martensitic stainless steel coating was prepared for by mechanical grinding, degreasing and etching in a solution

MW structures



Raw text from Steel coatings articles
Sentence normalisation
in a 30vol.%HCl solution
Returns XML Output
<MOLECULE><QUANTITY><PERCENT><CD>30</CD><NN>vol.</NN><NN-PERCENT>%</NN-PERCENT></PERCENT></QUANTITY><OSCARCM><OSCAR-CM>HCl</OSCAR-CM></OSCARCM>
Extract tagged molecule and quantity pairs
Molecule:HCl Quantity: 30vol.%
Verify extracted molecules using PubChempy
Molecule:HCl (verfied chemical name) Quantity: 30vol.%
Returns MW Structures
Molecule:HCl Quantity: 30vol.%

AMCPE Structures


Raw text from Steel coatings articles
Sentence normalisation
Returns XML Output
To generate a Ni–P/Ni–B duplex coating, the surface of the HVOF-sprayed 420 martensitic stainless steel coating was prepared for plating by mechanical grinding, acetone degreasing and etching in a 30 vol.%HCl solution for 1 min.
....<TimePhrase><IN-FOR>for</IN-FOR><CD>1</CD><NN-TIME>min</NN-TIME></TimePhrase>
Pass through nltk POS tagger
Extract words with VB/VBZ tag.
Returns Action words.
Actions: plating
Parse through Chemical Tagger
Extract molecule names
molecules: Ni-P/Ni-B, acetone, HCl
Extract tagged Temperatureand Time entities
Conditions: for 1min
Extract tagged quantities
Quantity: 30 vol.%
Extract process names using a predefined list of process names
Remaining sentence considered as extra
Extra: To generate a duplex coating, the surface of the HVOF-sprayed 420 martensitic stainless steel coating was prepared for by mechanical grinding, degreasing and etching in a solution

Preparation of Dataset

Consolidated dataset

Since Chemical Tagger wasn't giving optimum results we incorporated the help of more open-source libraries like **ChemData extractor**[18], **Quantulum**[19] and **AllenNLP**[20] to extract the desired entities. We wrote heuristics and used the libraries like to extract the required entities from raw coatings text.

The following example shows the output generated from each of the libraries mentioned above.

Sentence: To generate a Ni–P/Ni–B duplex coating, the surface of the HVOF-sprayed 420 martensitic stainless steel coating was prepared for plating by mechanical grinding, acetone degreasing and etching in a 30 vol.% HCl solution for 1 min.

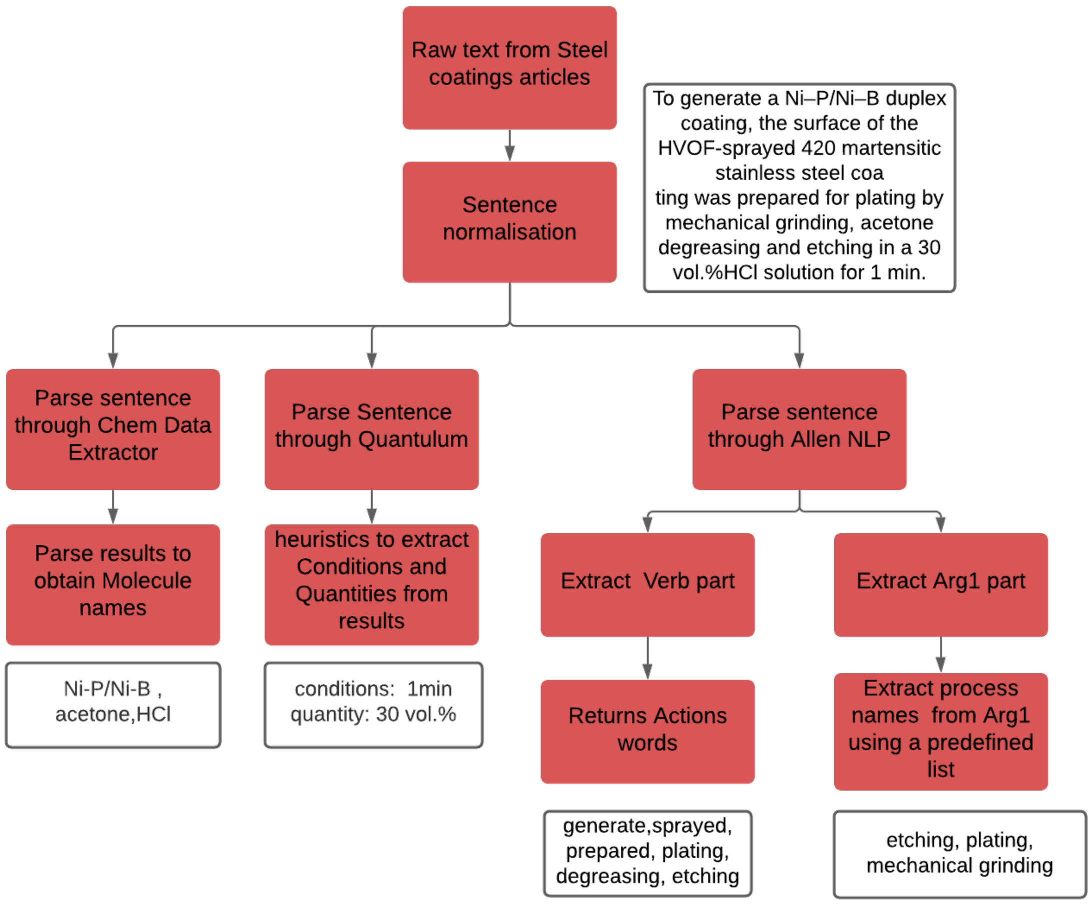
ChemData extractor: Ni-P/Ni-B, acetone, HCl
Allen NLP:
Verb: generate, sprayed, prepared, plating, degreasing, etching
Arg0: by mechanical grinding
Arg1: a Ni – P / Ni – B duplex coating , the surface of the HVOF - sprayed 420 martensitic stainless steel coating was prepared for **plating** by **mechanical grinding** , acetone **degreasing** and etching in a 30 vol.%HCl solution for 1 min, the surface of the HVOF - sprayed 420 martensitic stainless steel coating, for 1 min|in a 30 vol.%HCl solution
Arg2: -
Quantulum: 420, 30 vol.%, 1 min
Obtained Entities:

[18] http://chemdataextractor.org/

[19] https://github.com/marcolagi/quantulum

[20] https://github.com/allenai/allennlp

Molecules: Ni-P/Ni-B, acetone, HCl
Quantity: 30 vol.%
Actions:generate, sprayed, prepared, plating, degreasing, etching
Process: etching, plating mechanical grinding
Conditions: 1 min
Consolidated dataset (AMCPW)

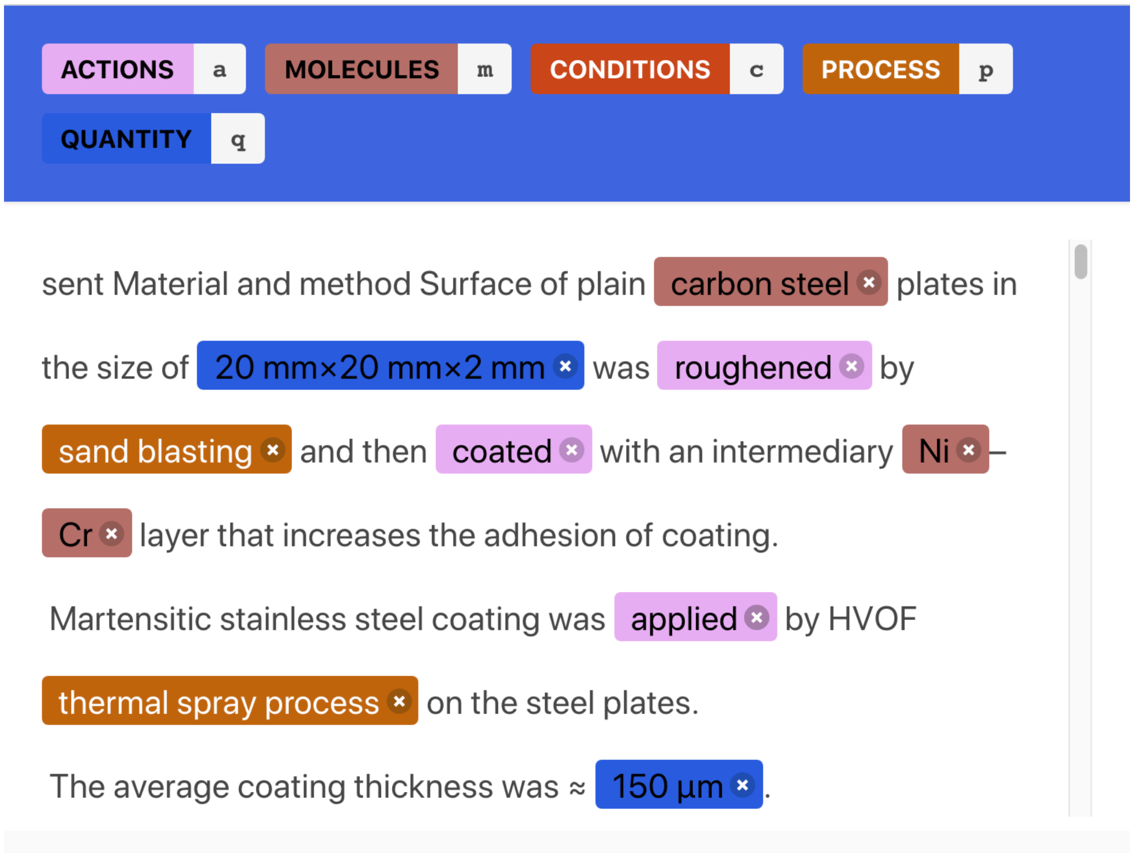


Raw text from Steel coatings articles
Sentence normalisation
To generate a Ni–P/Ni–B duplex coating, the surface of the HVOF-sprayed 420 martensitic stainless steel coa ting was prepared for plating by mechanical grinding, acetone degreasing and etching in a 30 vol.%HCl solution for 1 min.
Parse sentence through Chem Data Extractor
Parse Sentence through Quantulum
Parse sentence through Allen NLP
Parse results to obtain Molecule names
heuristics to extract Conditions and Quantities from results
Extract Verb part
Extract Arg1 part
Ni-P/Ni-B , acetone,HCl
conditions: 1min quantity: 30 vol.%
Returns Actions words
Extract process names from Arg1 using a predefined list
generate,sprayed, prepared, plating, degreasing, etching
etching, plating, mechanical grinding

Manual Annotation of Dataset

Since there were some errors in the extraction process, we manually annotated about 100 files using **Doccano**[21], an open source library for text annotation. The number of instances for each entity type was as follows:

Actions: 2107
Molecules: 954
Quantities: 439
Processes: 242
Conditions: 198



sent Material and method Surface of plain carbon steel plates in the size of 20 mm×20 mm×2 mm was roughened by sand blasting and then coated with an intermediary Ni–Cr layer that increases the adhesion of coating.

Martensitic stainless steel coating was applied by HVOF thermal spray process on the steel plates.

The average coating thickness was ≈ 150 µm.

The dataset was skewed with Actions and Molecules having the most number of entities. The dataset thus formed will be referred to as the **'Annotated Coatings dataset'** from here on.

[21] http://doccano.herokuapp.com/

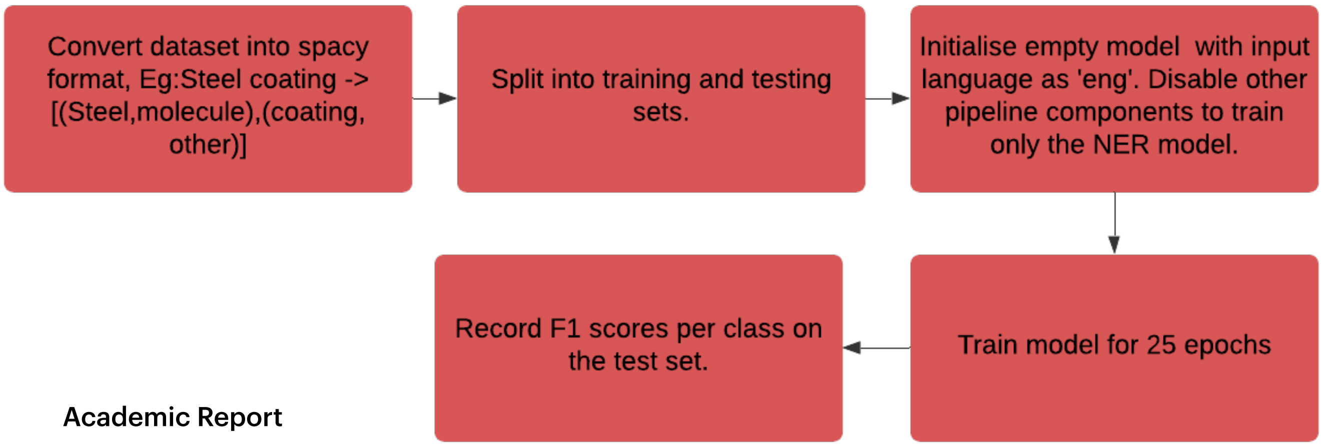
Exploration Work

NER Tagging

We now start with our task of training a custom NER model on the **Annotated Coatings Dataset**. We experiment with **Spacy NER**, **CRF Suite** and a **Seq2Seq model** using Keras.

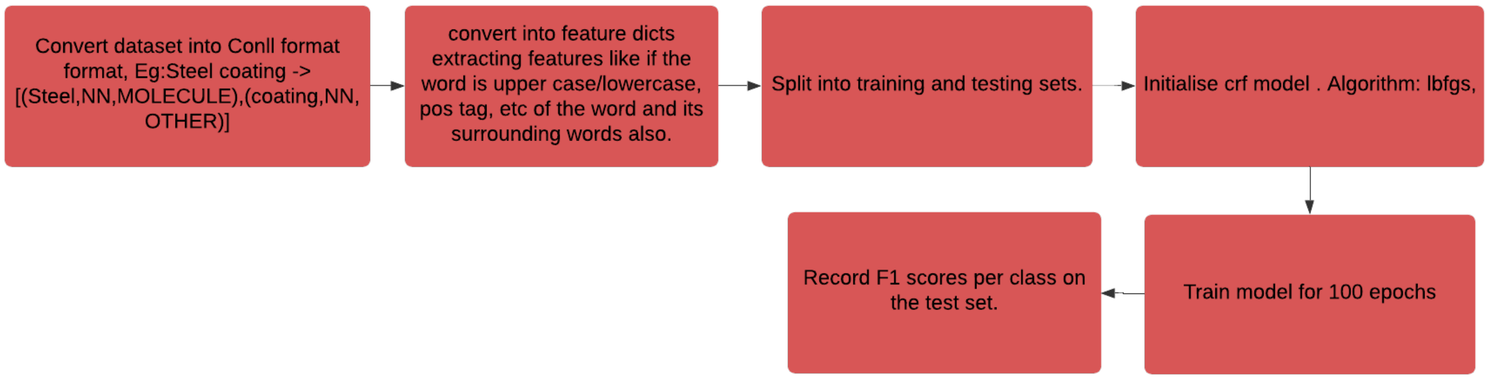
Spacy NER: Spacy provides 'empty' models for training of custom NER models. The model underneath preprocesses and trains using pipelines with an instance passing through a tokenizer, parsers, etc before being used for training. We made sure to disable other pipeline components before training so that only a NER model is trained. A consolidated **F1 score of 0.83** was achieved against a baseline of 0.69.

Class name	F1 score
Actions	0.64
Molecules	0.57
Quantity	0.53
Process	0.50
Conditions	0.53
Overall	0.83


Convert dataset into spacy format, Eg:Steel coating -> [(Steel,molecule),(coating, other)]
Split into training and testing sets.
Initialise empty model with input language as 'eng'. Disable other pipeline components to train only the NER model.
Train model for 25 epochs
Record F1 scores per class on the test set.

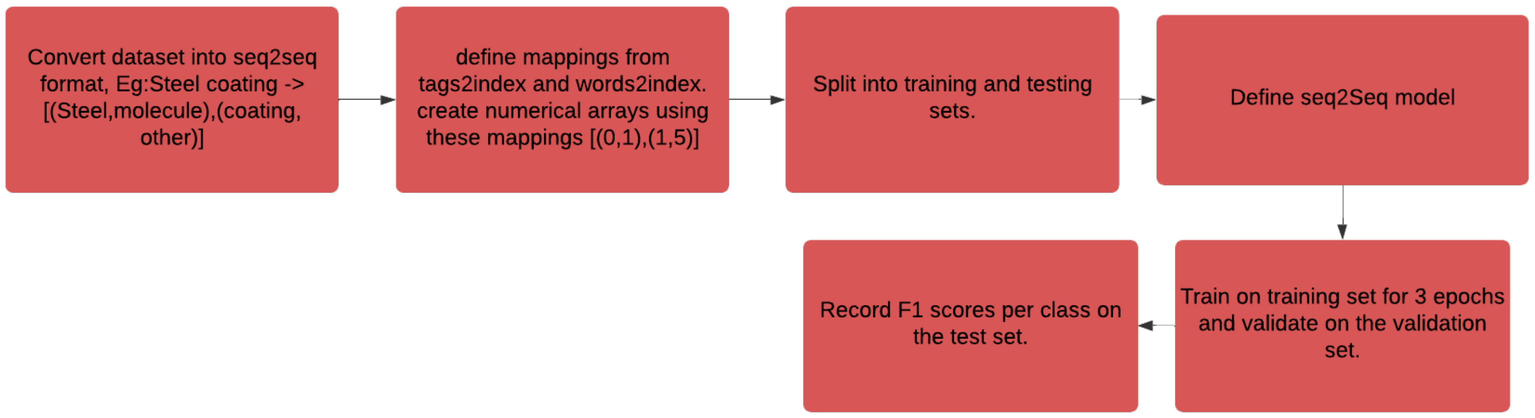
CRF Suite: Sklearn provides a wrapper for Crf Suite, an implementation of conditional random fields. Crf's are good in considering the context into account during the training process. The algorithm used is similar to gradient descent. For each word and its neighboring words in the training instance we created feature arrays by extracting features like POS tags, whether the word is in upper case or lower case, etc. We got a consolidated **F1 score of 0.88** against a baseline of 0.65.

Class name	F1 score
Actions	0.95
Molecules	0.43
Quantity	0.13
Process	0.167
Conditions	0.00
Overall	0.88


Convert dataset into Conll format format, Eg:Steel coating -> [(Steel,NN,MOLECULE),(coating,NN, OTHER)]
convert into feature dicts extracting features like if the word is upper case/lowercase, pos tag, etc of the word and its surrounding words also.
Split into training and testing sets.
Initialise crf model . Algorithm: lbfgs,
Train model for 100 epochs
Record F1 scores per class on the test set.

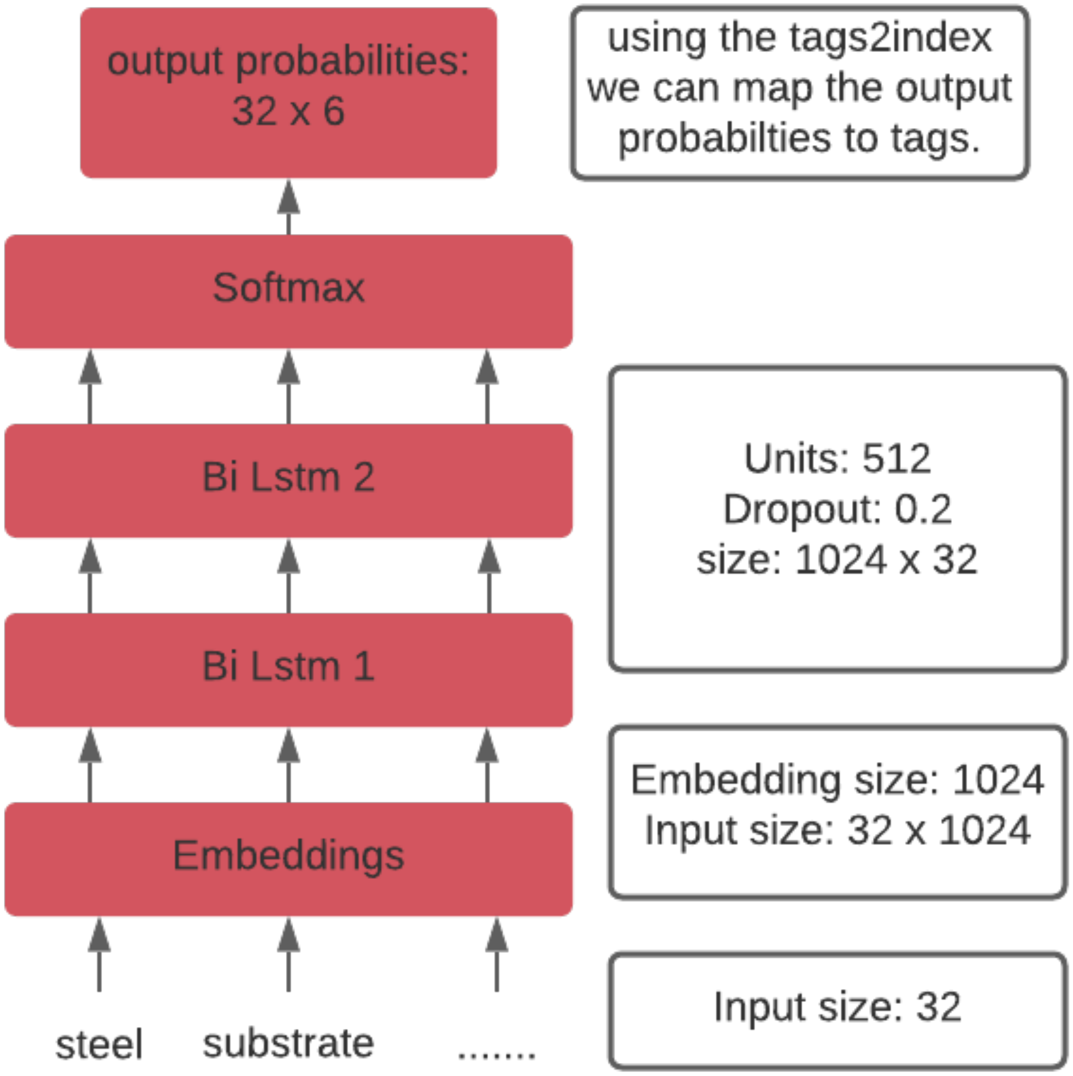
Seq2Seq using Keras on Tensorflow: We also experimented using deep learning techniques by implementing a Seq2Seq Model using Keras. The network was a Bi-directional LSTM with an added Elmo embedding layer as the input. We set a non zero value for dropout to ensure that the model does not overfit on our dataset. Since we require a lot of data for deep learning techniques, the model couldn't perform well and gave a consolidated **F1 score of 0.68**.

Class name	F1 score
Actions	0.93
Molecules	0.64
Quantity	0.0
Process	0.0
Conditions	0.0
Overall	0.68

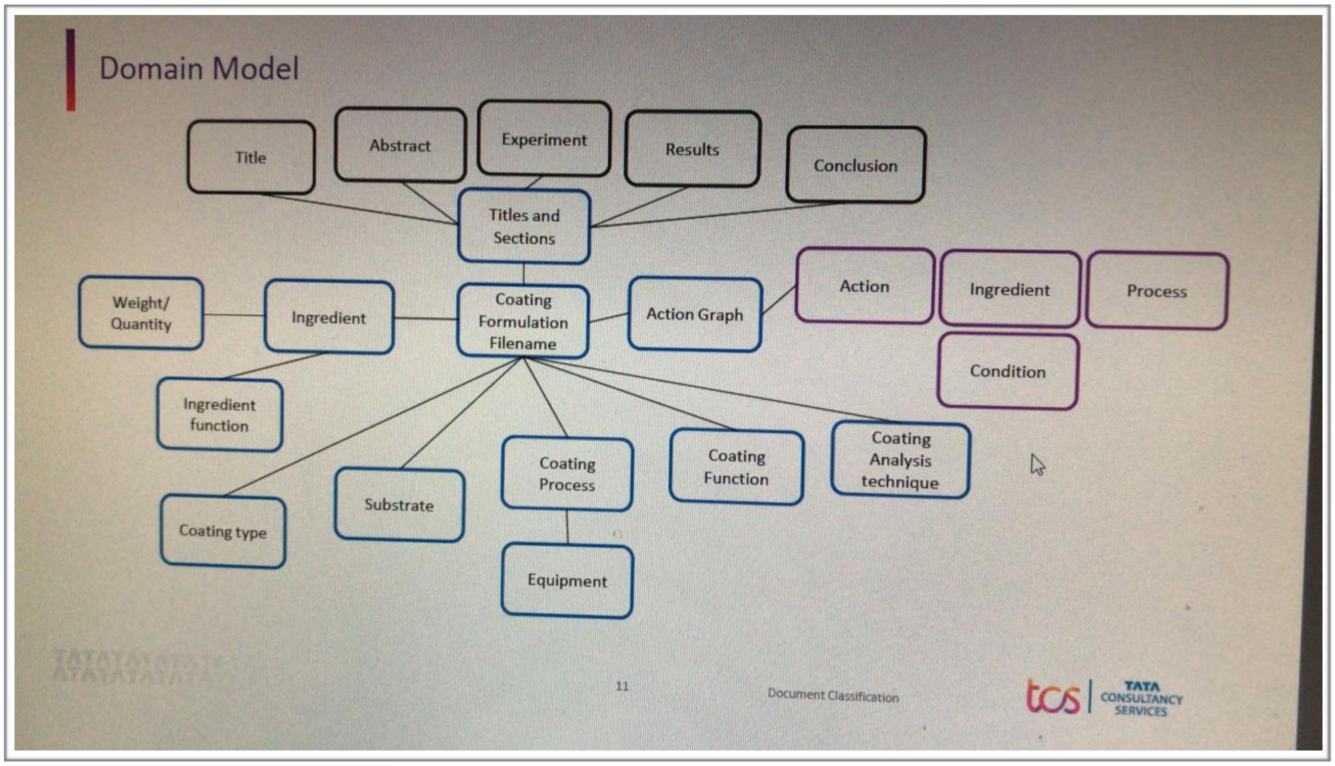

Convert dataset into seq2seq format, Eg:Steel coating -> [(Steel,molecule),(coating, other)]
define mappings from tags2index and words2index. create numerical arrays using these mappings [(0,1),(1,5)]
Split into training and testing sets.
Define seq2Seq model
Train on training set for 3 epochs and validate on the validation set.
Record F1 scores per class on the test set.

We used the trained **Spacy NER model** as our final model as it was able to detect all types of entities with good accuracy and was able to handle the skewness in the dataset well.

SEQ2SEQ MODEL DESCRIPTION:



Domain Model



Search and Annotation Tool

Our final aim was to extract the entities given in the domain model. Till now we were able to extract the basic entities. We figured out that the basic entities along with some keywords could help us populate our domain model. For example, To identify 'substrates' in a coating process, we could search for the sentences where the word 'substrate' would occur. Then we could manually annotate the name of the substrate in that sentence. This was the motivation behind the Search cum Annotation tool.

Search Tool: The search tool provides support for firing queries in different formats to the end user. The motivation behind this was that the end user could query and observe trends and understand how text was structured in the dataset. Once identified the user can fire specific queries that would return results closest to what he/she wants.

Queries Supported: The user can write queries in 4 different formats
Boolean: It consists of word(s) and there different combinations
For example, The user can search for
A single word: 'substrate'
Multiple words: 'stainless steel'
Multiple word combinations: 'stainless|carbon steel' (here it searches for the combinations 'stainless steel' and 'carbon steel'

Field: In addition to words, one can also search for a particular lemma, any predefined entity from molecules, actions, processes, quantities and conditions or words having a specific POS (Part of speech) tag. The user can fire single, multiple and multiple word combinations as in the case of Boolean queries.
For Example: The user can search for

Single/Multiple/Mutiple word combos: ‘word=substrate’/ ‘word=stainless steel’/ ‘word= stainless|word=carbon word=steel/ ‘lemma=coating/ tag=‘NN’/ Using|by entity=Process’

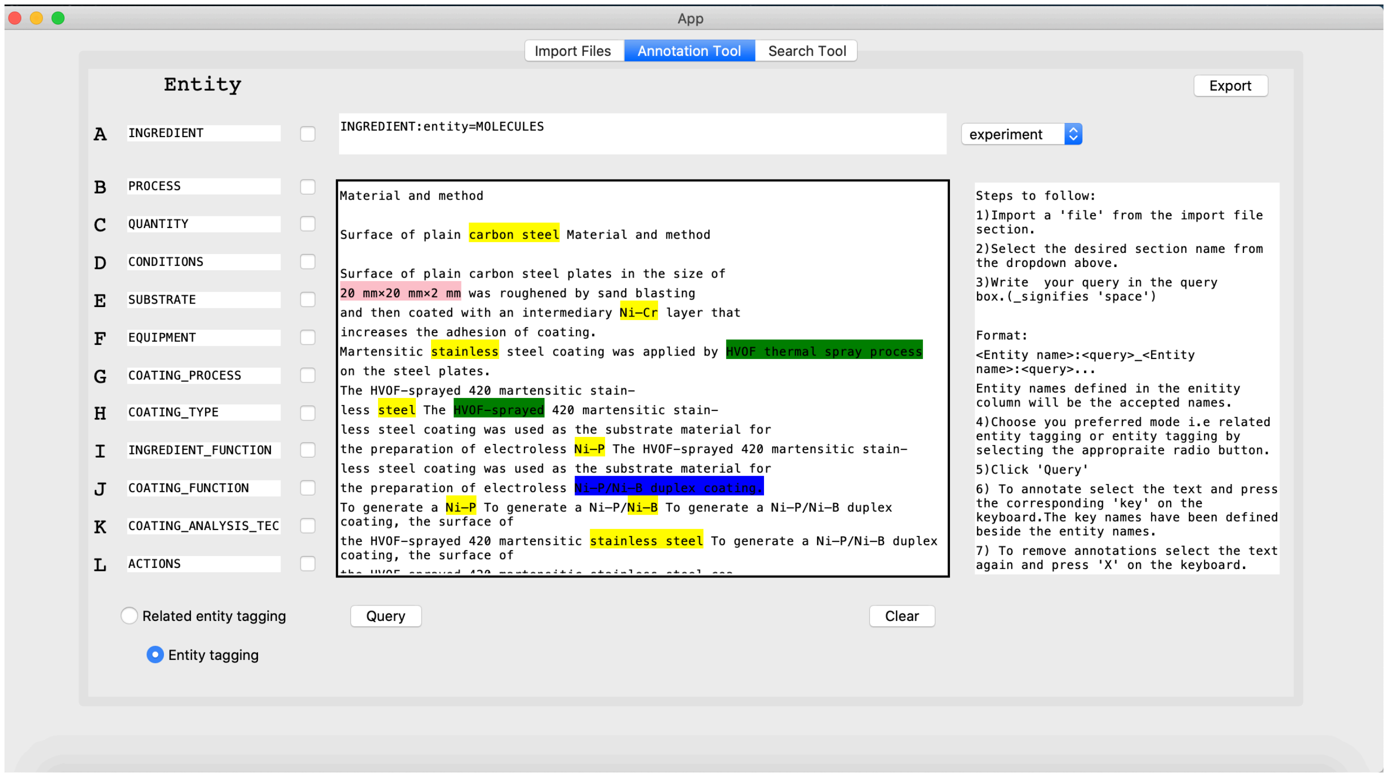
Multiple pairs: entity=Molecules entity=Process (searches for sentence with occurrence of either one of the entities)

Capture Groups: These enable us to capture any field query in a variable.
For Example:
Ingredient: entity=Molecules process_name:entity=process
Substrate:word=stainless|word=carbon word=steel

Sequential: In the above mentioned queries, the order of occurrence of words wasn’t considered, but in sequential queries, one can search for Boolean/Field/Capture groups in a particular order.
For example:
stainless, steel (Here only those sentences will be returned who have the word stainless followed by steel, so the order will be strictly followed)

Annotation Tool: The annotation tool provides support to search and annotate predefined entities that are outlined in the domain model. Support is provided to the user to annotate standalone entities or define relationships between them by annotating them in sequence. For example, the user can annotate names of substrates or can annotate molecule-quantity pairs. The annotations can be exported in CSV format.

Annotation Tool



Entity

A	INGREDIENT	☐
B	PROCESS	☐
C	QUANTITY	☐
D	CONDITIONS	☐
E	SUBSTRATE	☐
F	EQUIPMENT	☐
G	COATING_PROCESS	☐
H	COATING_TYPE	☐
I	INGREDIENT_FUNCTION	☐
J	COATING_FUNCTION	☐
K	COATING_ANALYSIS_TEC	☐
L	ACTIONS	☐

```
INGREDIENT:entity=MOLECULES
```

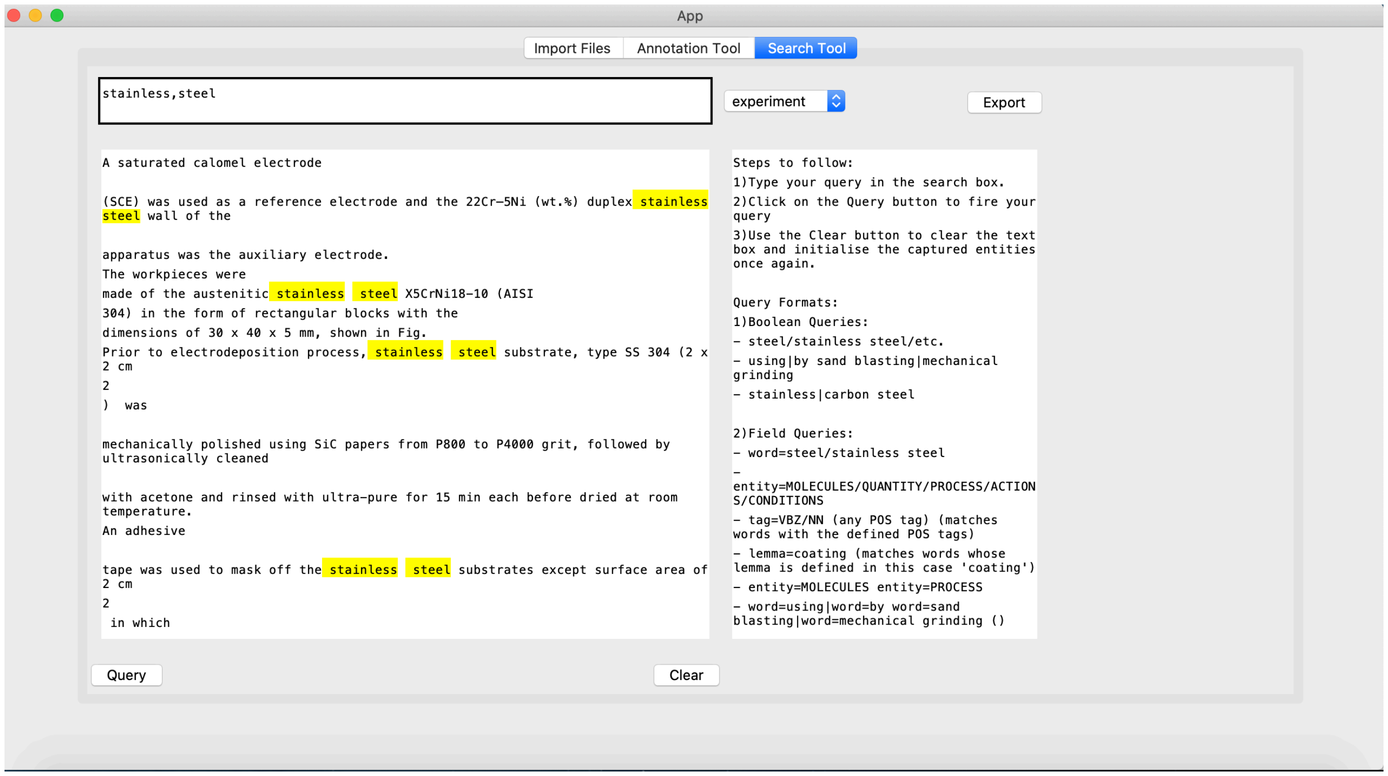
experiment

```
Material and method

Surface of plain carbon steel Material and method

Surface of plain carbon steel plates in the size of
20 mm×20 mm×2 mm was roughened by sand blasting
and then coated with an intermediary Ni–Cr layer that
increases the adhesion of coating.
Martensitic stainless steel coating was applied by HVOF thermal spray process
on the steel plates.
The HVOF–sprayed 420 martensitic stain–
less steel The HVOF–sprayed 420 martensitic stain–
less steel coating was used as the substrate material for
the preparation of electroless Ni–P The HVOF–sprayed 420 martensitic stain–
less steel coating was used as the substrate material for
the preparation of electroless Ni–P/Ni–B duplex coating.
To generate a Ni–P To generate a Ni–P/Ni–B To generate a Ni–P/Ni–B duplex
coating, the surface of
the HVOF–sprayed 420 martensitic stainless steel To generate a Ni–P/Ni–B duplex
coating, the surface of
```

```
Steps to follow:
1)Import a 'file' from the import file
section.
2)Select the desired section name from
the dropdown above.
3)Write  your query in the query
box.(_signifies 'space')

Format:
<Entity name>:<query>_<Entity
name>:<query>...
Entity names defined in the enitity
column will be the accepted names.
4)Choose you preferred mode i.e related
entity tagging or entity tagging by
selecting the appropraite radio button.
5)Click 'Query'
6) To annotate select the text and press
the corresponding 'key' on the
keyboard.The key names have been defined
beside the entity names.
7) To remove annotations select the text
again and press 'X' on the keyboard.
```

○ Related entity tagging

● Entity tagging



```
stainless,steel
```

experiment

```
A saturated calomel electrode

(SCE) was used as a reference electrode and the 22Cr–5Ni (wt.%) duplex stainless
steel wall of the

apparatus was the auxiliary electrode.
The workpieces were
made of the austenitic stainless  steel X5CrNi18–10 (AISI
304) in the form of rectangular blocks with the
dimensions of 30 x 40 x 5 mm, shown in Fig.
Prior to electrodeposition process, stainless  steel substrate, type SS 304 (2 x
2 cm
2
)  was

mechanically polished using SiC papers from P800 to P4000 grit, followed by
ultrasonically cleaned

with acetone and rinsed with ultra–pure for 15 min each before dried at room
temperature.
An adhesive

tape was used to mask off the stainless  steel substrates except surface area of
2 cm
2
 in which
```

```
Steps to follow:
1)Type your query in the search box.
2)Click on the Query button to fire your
query
3)Use the Clear button to clear the text
box and initialise the captured entities
once again.

Query Formats:
1)Boolean Queries:
- steel/stainless steel/etc.
- using|by sand blasting|mechanical
grinding
- stainless|carbon steel

2)Field Queries:
- word=steel/stainless steel
-
entity=MOLECULES/QUANTITY/PROCESS/ACTION
S/CONDITIONS
- tag=VBZ/NN (any POS tag) (matches
words with the defined POS tags)
- lemma=coating (matches words whose
lemma is defined in this case 'coating')
- entity=MOLECULES entity=PROCESS
- word=using|word=by word=sand
blasting|word=mechanical grinding ()
```

Query Tool

Code Base References

https://spacy.io/usage/training#ner
https://medium.com/@manivannan_data/how-to-train-ner-with-custom-training-data-using-spacy-188e0e508c6
https://sklearn-crfsuite.readthedocs.io/en/latest/tutorial.html
https://github.com/nxs5899/Named-Entity-Recognition_DeepLearning-keras/blob/master/Named_Entity_Recognition_ELMo.ipynb
https://github.com/creme-ml/creme
https://github.com/marcolagi/quantulum
http://chemdataextractor.org/
https://github.com/allenai/allennlp
https://docs.python.org/3/library/tk.html