Using Conditional Random Fields to Detect Catchphrases in Legal Documents

Yogesh H. Kulkarni
Consultant, RightSteps Consultancy, Pune, India.
yogeshkulkarni@yahoo.com

Rishabh Patil
Engineer, RightSteps Consultancy, Pune, India.
rishabh@rightstepsconsultancy.com

Srinivasan Shridharan
Founder, RightSteps Consultancy, Pune, India.
srini@rightstepsconsultancy.com

ABSTRACT

"Common Law System" practiced in India refers to statute as well as precedent to form judgments. As number of cases are increasing rapidly, an automatic precedent retrieval system is desirable. One of the key requirements of such information retrieval system is to pre-populate database of prior cases with catchphrases for better indexing and faster, relevant retrieval. This paper proposes an automatic catchphrases prediction for cases for the same. The problem catchphrase detection has been modeled as "custom named entity recognition (NER) using conditional random fields (CRF)". CRF is trained with pairs of prior cases and their respective catchphrases, the gold standards. The model is, then used to predict catch-phases of unseen legal texts. Towards end, this paper demonstrates efficacy of the proposed system using practical data-set.

CCS CONCEPTS

• **Computing methodologies → Information extraction**; • **Information systems → Probabilistic retrieval models**;

KEYWORDS

Information Retrieval, Conditional Random Fields, Named Entity Recognition, Legal, Text Mining, Natural Language Processing.

ACM Reference Format:
Yogesh H. Kulkarni, Rishabh Patil, and Srinivasan Shridharan. 2017. Using Conditional Random Fields to Detect Catchphrases in Legal Documents. In *Proceedings of Forum for Information Retrieval Evaluation (Fire-IRLeD'17).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.475/123_4

1 INTRODUCTION

Indian judicial system, like many in the other parts of the world, is based on whats called "Common Law System" in which both, written law (called "statutes") and prior cases (called "precedent") are given equal importance while forming the judgment. Such system brings uniformity of the legal decisions across similar situations. With number of cases increasing day by day, it has become humanly impossible to search relevant past cases for a particular topic. Automatic Precedent Retrieval System (APRS) is the need of the hour. As more and more cases are coming in the digital form, text mining has found immense importance for developing APRS.

Fire-IRLeD'17, Dec 2017, IISc, Bangalore, India

ACM ISBN 123-4567-24-567/08/06…$XX.00
https://doi.org/10.475/123_4

Court cases, judgments, legal texts are typically long and unstructured, making it hard to query relevant information from them, unless someone goes through them manually and vigilantly. Looking at the volume of legal text to be processed, it is desirable to have automatic system that detect key concepts, catchphrases in the legal texts.

The aim of this paper is to propose automatic catchphrase detection-prediction system for legal text. It uses training data comprising of pairs of text and respective catchphrases, the gold standard, prepared manually by legal experts. The proposed system builds probabilistic model based on this training data, which, in turn, can predict the catchphrases in the unseen legal texts.

The contributions made in this paper are as follows:

(1) A novel method to prepare training data needed for CRF.
(2) Feature engineering for better results with CRF

The paper has been structured as follows: in the following section 2 the catchphrases detection task has been described in details, as definition of the problem. In section 3, structure of the training data has been explained. Next, the proposed system is elaborated in Section 4. It describes preparation of CRF training data-set and feature engineering adapted for this custom named entity recognition (NER) methodology. Section 5 discusses the findings drawn from this work.

2 TASK DEFINITION

Catchphrases are short phrases from within the text of the document. Catchphrases can be extracted by selecting certain portions from the text of the document[2]. The data-set provided consists of legal texts and their respective catchphrases, along with test documents for which the catchphrase needs to be extracted-predicted.

3 DATA-SET

Fire-2017 [2] dataset contains following directories:

(1) Train_docs : contains 100 case statements, $case\_ < i > \_statement.txt$ where $i = 0 \rightarrow 99$. Sample document looks like "R.P. Sethi, J. 1. Aggrieved by the determination of annual …ultimate result.".
(2) Train_catches: contains the gold standard catchwords for each of these 100 case statements, $case\_ < i > \_catchwords.txt$ where $i = 0 \rightarrow 99$. Sample document looks like "Absence, Access, Accident, Account, …Vehicle, Vehicles".
(3) Test_docs: contains 300 test case statements, similar to Train_docs, $case\_ < i > \_statement.txt$ where $i = 100 \rightarrow 399$.

4 SYSTEM DESCRIPTION

4.1 Preprocessing

Each of the training statements were tokenized into a list. Their Parts-of-Speech (POS) tags were generated using python nltk [1] library. Another sequence of custom NER tagging was made by referring to token list and given catchphrases. B-LEGAL and I-LEGAL tags were employed for Begin and Intermediate of the catchphrases respectively and O for other tokens. So training data file looked like:

in	IN	O
the	DT	O
year	NN	O
1987	CD	O
and	CC	O
that	IN	O
property	NN	B-LEGAL
had	VBD	O
extensive	JJ	O
national	JJ	B-LEGAL
highway	NN	I-LEGAL
frontage	NN	O

Table 1: Training data with primary features

There are 3 columns for each token.

(1) The word itself (e.g. property);
(2) POS associated with the word (e.g. NN);
(3) Custom NER tag (e.g. B-LEGAL);

Each test statement was also tokenized into a list and its POS tags were generated using nltk [1], so testing data file looked like:

appeals	NN
the	NNS
high	DT
court	JJ
accepted	NN
the	VBD
view	DT

Table 2: Testing data with primary feature

There are 2 columns for each token.

(1) The word itself (e.g. appeals);
(2) POS associated with the word (e.g. NN);

4.2 Modeling

The problem of detecting catchphrases was modeled as customized NER. POS and custom NER tagging performed during pre-processing stage were used to form secondary features. These were used in building CRF model. CRF++ [3] toolkit was used. Salient secondary features developed were:

(1) Unigrams:
  (a) Previous 3 tokens, current token and next 3 tokens
  (b) Previous 3 POS tags, current POS tag and next 3 POS tags
(2) Bigram tokens

CRF model was generated using:

```
crf_learn  template_file  train_file  model_file
```

The generated model file was then used to predict from test data:

```
crf_test  −v1 −m model_file  test_files
```

With $-v1$ option the highest probability is shown as:

Rockwell	NNP	B	B/0.992465
International	NNP	I	I/0.979089
Corp.	NNP	I	I/0.954883

Table 3: Sample results with probabilities

4.3 Results

The CRF++ model was used to predict custom NER tags from the given testing data as:

case_102_statement	notification:0.990733,tax:0.7341635
case_103_statement	prevention of corruption:0.9988746666666667
case_104_statement	natural justice:0.7491485,appeal:0.708494
case_105_statement	seniority:0.997623,legislation:0.994512,appointing

Table 4: Submission file

There far less catchphrase words compared to total number of words in the documents. Thus, accuracy is not a good metrics to measure the performance of this prediction. "Precision" and "Recall" values on the testing data-set came out to be as follows:

Mean Average Precision	Overall Recall
0.47923074	0.2476876075

Table 5: Results conveyed by FIRE[2]

5 CONCLUSIONS

In this paper, a brief overview of Automatic Catchphrases Prediction System was presented to extract catchphrases from legal texts. Accuracy was impacted as some of the training samples had very few catchphrases. Various apporaches/toolkits were tried but it was found that the problem of catchphrase detection needs to be modeled as sequential probabilistic labeling problem rather than a simple linear classification problem. CRF algorithm was chosen with primary features as POS and custom NER tags and numerous secondary features representing the context. As a future work, if sufficient gold standard data is available, one can explore more sophisticated techniques such as Long Short Term Memory networks (LSTM), where custom features need not be provided but get generated internally.

VITAE

Yogesh H. Kulkarni works as Data Science Consultant and Trainer. Profile: https://www.linkedin.com/in/yogeshkulkarni/
Rishabh Patil works as Data Engineer. His profile is at https://www.linkedin.com/in/rishabh-patil-256a25124/.
Srinivasan Shridharan is Data Scientist and entrepreneur. Profile: https://www.linkedin.com/in/srinivasan-shridharan-08a86a6/.

ACKNOWLEDGMENTS

Wish to thank Ankur Parikh, a keen researcher of Deep Learning and NLP, for discussions.

REFERENCES

[1] Edward Loper Bird, Steven and Ewan Klein. 2009. Natural Language Toolkit. (Sep 2009). http://www.nltk.org/
[2] Information Retrival Society of India. 2017. Forum for Information Retrieval Evaluation. (Dec 2017). https://sites.google.com/view/fire2017irled/track-description
[3] Taku. 2017. CRF++: Yet Another CRF toolkit. (Sep 2017). https://taku910.github.io/crfpp/