Court Reversals: Predicting the decision of Circuit and Supreme Court of United States

Amanpreet Singh[1], Sharan Agrawal[2], Simranjyot Singh Gill[3], and Karthik Venkatesan[4]

Courant Institute of Mathematical Sciences, 251 Mercer St, New York, NY 10012 USA
[1]amanpreet@nyu.edu,[2]sharan.agrawal@nyu.edu,[3]ssg441@nyu.edu,[4]kv730@nyu.edu

Abstract. Every year more than 300,000 civil and criminal cases are heard in the district courts all over the US. Less than 5% of these cases are appealed and heard in circuit courts. For most of the cases, the circuit court either affirms the decision of the district court or reverses it. Out of the cases heard in circuit courts, only about 2% are heard in the Supreme Court. The Supreme Court again can again either affirms the decision of the circuit court or reverses it. In this project, we have attempted to predict reversal of the district court opinion by the circuit court and the circuit court opinion by the Supreme Court, using the summarized opinion of the district court. We trained a wide variety of classifiers on our data and also used dimensionality reduction and oversampling techniques to improve the classifier.

Keywords: district court, circuit court, supreme court, reversal, scotus

1 Introduction

In the United States, the federal court system has three main levels: district courts (the trial court), circuit courts which are the first level of appeal, and the Supreme Court of the United States (which will referred to as SCOTUS in this paper), the final level of appeal in the federal system. The district courts are the general trial courts of the federal court system. All cases are heard in district court and a verdict is announced. These district cases sometimes are appealed in the circuit court where a decision is made to affirm or reverse the decision of district court partly or completely. After circuit court, the final appeal can be made in Supreme Court of United States which decides to affirm or reverse the decision of the circuit court. In this paper, we tried to build two classifier models, one for district court to circuit court and another for circuit court to SCOTUS. We will delineate what plays as an important factor when a court decision is made. Finally, we will try to create a general model for court decision predictions from the two classifiers we have created.

2 Data

2.1 District to Circuit

Our data consisted of 3 separate data sets:

1. Raw records of approximately 750,000 U.S Circuit Court Opinions from 1880 to 2013 (Both published and unpublished). This complete and comprehensive dataset was cleaned, parsed and processed to find matching district court case using case name, circuit number and date published.
2. Raw text of approximately 280,000 District Court opinions from 1924 to 2013. This dataset, even though incomplete, is a strong representative of the decisions of the district court. A CSV file describing these district court cases containing case metadata. Ngrams were filtered out using the raw text files containing the decision texts.
3. A DTA file containing information about 4096 judges. This information was used to filter out district court judges from the CSV file and match them with the details mentioned in this file.

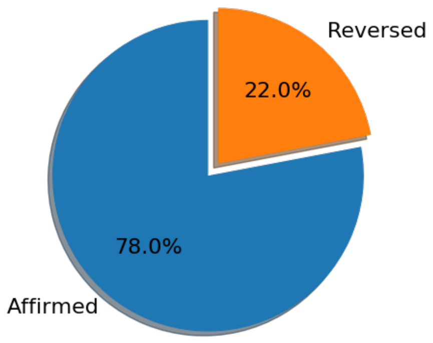

Fig. 1. Partition of Affirmed and Reversed cases in district corpus

2.2 Circuit to SCOTUS

Our data consisted of 2 separate data sets:

1. Circuit court data contains scraped html web pages from various sources for 387,000 cases, which have been cleaned to retain important information. We have generated 2-grams through 4-grams from these case records.
2. Circuit Court- Supreme Court merged data-set: This data-set comprises of 4000 circuit court cases which were heard by supreme court and also have target variable, judges biography and few additional features merged in.

3. As opposed to the district to circuit court data, the number of reversals of circuit court decisions by SCOTUS is higher as compared to the number of affirms.

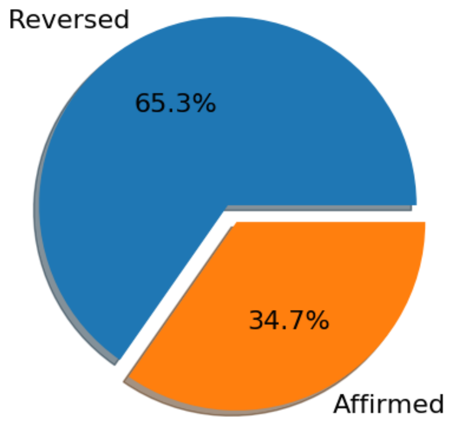

Fig. 2. Partition of Affirmed and Reversed cases in circuit corpus

3 Preprocessing Pipeline

3.1 District to Circuit

The preprocessing for the District to Circuit consisted of the following steps:

3.1.1 Matching District Cases to corresponding Circuit Cases As explained in the last section, our data was divided into 2 separate datasets. There was a corpus of circuit cases and raw text files of district cases. We used case names to link the 2 datasets. We leveraged the CSV file containing district court metadata, to get the date of the case and the case name. We then parsed the raw decision texts of all the circuit cases in the next 5 years to get case names and tried to find a match with the case name found in the csv. We considered a similarity score of 0.95 equivalent to a match in the case names as the exact matches were very rare. We handled the multiple matches by increasing the similarity score and matching the district until only a unique match was found. Out of the 280,000 district cases, we found around 40,000 matches to the corresponding circuit cases. However, 25,000 of these were unpublished and were not present in the Affirm-Reverse pickle, which indicated whether a particular circuit court case was affirmed or reversed. Thus, our dataset now contained 15,000 cases which were matched to their corresponding Circuit cases.

3.1.2 Matching Judges to District Cases

For matching the judges we used the cases matched in the last part from the district cases metadata CSV file and the DTA file which contained the information about the judges. The first step was to filter out all the district court judges, this information was given clearly in the DTA. In the next step, we had to match the cases to the corresponding judge biographies. The CSV consisted of names of the judges corresponding to each case. However, the names were not complete and in most of the cases only consisted of the last names. We tried to find exact match of the songername with the name of the judge given in the CSV in the first pass. If the exact match was not found, we considered a similarity score of 0.95 between the last name given in the DTA and CSV file as an exact match. Additionally, we matched the district court in which the case was heard (present in the CSV) and the district court of the judge under consideration. We also took into consideration the time frame in which the judge was appointed and made sure that the case was also within this time frame. After enforcing all the conditions about 6200 cases out of the 15000 that were matched to their circuit cases earlier were matched with their judge biography in the DTA file.

3.2 Circuit to SCOTUS

The preprocessing for the Circuit to SCOTUS consisted of the following steps:

3.2.1 Matching Circuit Cases to corresponding SCOTUS Cases

Our data was also divided into 2 separate datasets. There was a corpus of circuit cases html files which had been cleaned to retain important information in the form of raw text. This corpus was used to generate n-grams. Apart from this there was another dataset in the form of CSV file, which had merged and matched all the circuit court cases with the SCOTUS cases. This dataset consists of case-id, judges-biography and other important features. This dataset also contained the affirm/reverse variable which we were trying to predict. We used case-ids to link the 2 datasets. Our final dataset consisted of 4300 circuit court cases which were matched to their corresponding SCOTUS cases.

3.2.2 Dataset Generation

We tried two approaches to dataset generation.

1. We built one dataset with the all the cases upto a certain term as the training data and all the cases after the term as the test data.
2. Next, we sampled a proportion of cases from each term (0.75) as training data and rest (0.25) contributed to test data.

We found that using Method 2 for data generation did not yield good results suggesting that future cases might not be a good indicator of reversal of past cases. So we continued with the dividing of data based on particular term for the rest of our analysis, ie, Method 1.

4 Feature Engineering

4.1 District to Circuit

In district to circuit, there were three main kinds of features involved: Ngrams, Judges features and Word2Vec[5] embeddings. We parsed, reduced and engineered these features to provide us the best results.

4.1.1 NGrams Reversal prediction was mainly based on ngram feature. After filtering out the district cases with information about reversals, we generated 1-4 grams from the text of district cases. We used legal context free grammar for this purpose which is shown below:

$$\begin{aligned}
&\text{S} \Rightarrow \text{TWO} \mid \text{THREE} \mid \text{FOUR}\\
&\text{TWO} \Rightarrow \text{A N} \mid \text{N N N} \mid \dots\\
&\text{THREE} \Rightarrow \text{N N N} \mid \text{A A N} \mid \dots\\
&\text{FOUR} \Rightarrow \text{N C V N} \mid \text{A N N N} \mid \dots\\
&\text{A} \Rightarrow \text{JJ} \mid \text{JJR} \mid \text{JJS}\\
&\text{N} \Rightarrow \text{NN} \mid \text{NNS} \mid \text{NNP} \mid \dots\\
&\text{V} \Rightarrow \text{VB} \mid \text{VBD} \mid \text{VBG} \mid \dots
\end{aligned}$$

This ngrams were then filtered based on frequency. We took top $\frac{1}{8^{th}}$ most occurring and top $\frac{1}{8^{th}}$ least occurring ngrams from the total to reduce the dimensionality of the ngram feature as the middle ngrams are likely to have any effect.[1]. Stopwords and other common court terms were removed from the ngrams. Finally, ngrams were stored into files with their respective frequency to be used later.

4.1.2 Judges: Using the DTA file describing the information of the judges, we were able to filter out important features of each judge. We used the age, political affiliation, district and gender as additional judge features in our model. We combined these features with ngrams in the final stage.

4.1.3 Word2Vec Embeddings For state of art classification of text using convolutional neural networks[4] [9], we generated a word2vec model for court cases using gensim [6] library. Word2Vec model's vocabulary was created on district court training data and was trained on both district court and circuit court texts. Number of dimensions were set to 100 with minimum count of 15 and 50 iterations.

4.2 Circuit to SCOTUS

4.2.1 NGrams Refer to Section 4.1.1.

4.2.2 Judges Biography The Circuit Court Judges biography data was the first set of features that were included into the model. We tried to reflect the party affiliations of each judge and also the overall party affinity of the juding panel as a whole. For this we included variables for each judge telling whether they are Republican, Democrat or belong to another party. We also included the ratio of judges in the panel belonging to each party as a normalized score of the representation of each of the parties under consideration.

4.2.3 Case Characteristics Next, we tried to incorporate important characteristics of the case into the model. We started with adding the variables lower court disposition to indicate whether the case was affirmed or reversed when it was appealed from the district court to the circuit court. Next, we added the feature representing dissent amongst the district court judges while making the decision as we felt that cases with judgment which was not unanimous tend to be better chances of reversal. Finally, we added the issue area under which the case falls under. This was done to reflect the category of cases which might influence reversals. The areas included include 'Criminal Procedure','Civil Rights', 'First Amendment', 'Due Process', 'Privacy', 'Attorneys', 'Unions', 'Economic Activity', 'Judicial Power', 'Federalism', 'Interstate Relations', 'Federal Taxation', 'Private Action' and 'Miscellaneous'. Initially, we added the view of the case as conservative or liberal as a feature but later removed it fearing bias in the original classification.

5 Classifier Models

5.1 District to Circuit

After passing through Ngram and Judges pipeline and applying various techniques such as oversampling to overcome the effect of imbalanced classes, we trained a variety of classifiers on varied metrics to choose the one that performed best. Initially, we started from simple array of classifiers like LinearSVM and Multinomial Naive Bayes to see how well ngrams predict the reversals on the baseline model. Further, we moved on the ensembles, in order to further reduce the effect of imbalanced classes in the dataset. We trained Random Forest and Gradient boosting with regression trees classifiers. However, the results obtained from these baseline models were not very promising. We then decided to try a different approach and shifted to Convolutional Neural Networks with Word2Vec embeddings [4], which produced much better results as compared to the baseline models.

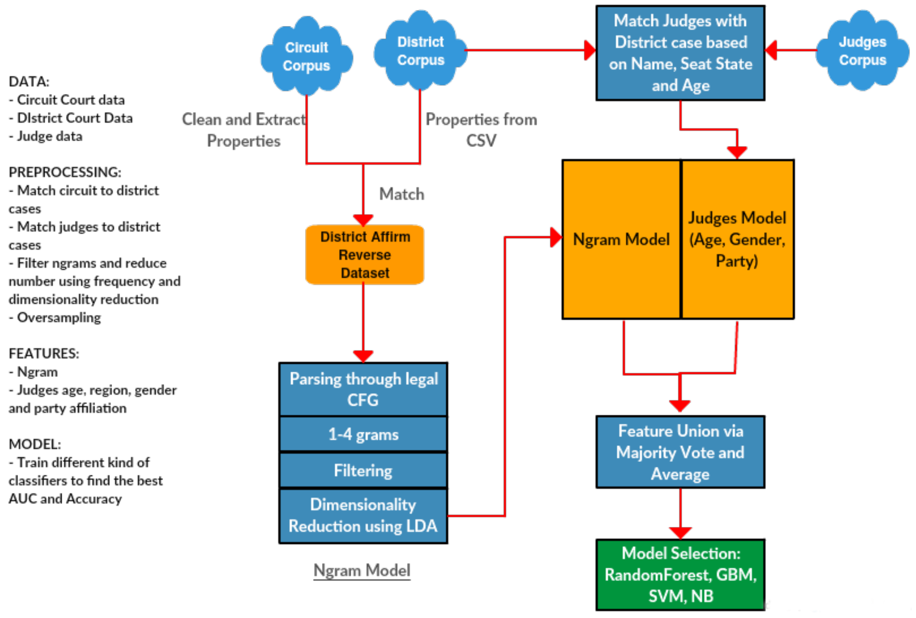
DATA:
- Circuit Court data
- DIstrict Court Data
- Judge data
PREPROCESSING:
- Match circuit to district cases
- Match judges to district cases
- Filter ngrams and reduce number using frequency and dimensionality reduction
- Oversampling
FEATURES:
- Ngram
- Judges age, region, gender and party affiliation
MODEL:
- Train different kind of classifiers to find the best AUC and Accuracy
Circuit Corpus
District Corpus
Clean and Extract Properties
Properties from CSV
Match
District Affirm Reverse Dataset
Parsing through legal CFG
1-4 grams
Filtering
Dimensionality Reduction using LDA
Ngram Model
Match Judges with District case based on Name, Seat State and Age
Judges Corpus
Ngram Model
Judges Model (Age, Gender, Party)
Feature Union via Majority Vote and Average
Model Selection: RandomForest, GBM, SVM, NB

Fig. 3. Pipeline diagram for district court to circuit court

5.1.1 Overcoming imbalance: As the reversed cases were in minority, most of the classifiers predicted affirm in almost all the samples and considered reversals as outliers to achieve high accuracy. To overcome this problem [3], we applied oversampling to increase the weight of reversal cases by duplication. Further, we applied LDA to reduce the dimensionality of feature matrix which becomes quite huge due to presence of 1-4 grams[8]. After application of these techniques, data became balanced and free from curse of dimensionality.

5.1.2 Training: We used the inbuilt dict vectorizer function of scikit learn to vectorize the feature value mappings of ngrams into set of vectors which can be given as input into various scikit learn classifiers. Finally, the vectorized ngrams and output vector of label classes "Affirmed" and "Reversed" is given as input into fit function of the classifier. This is the general way of training we used for baseline classifiers and ensembles.

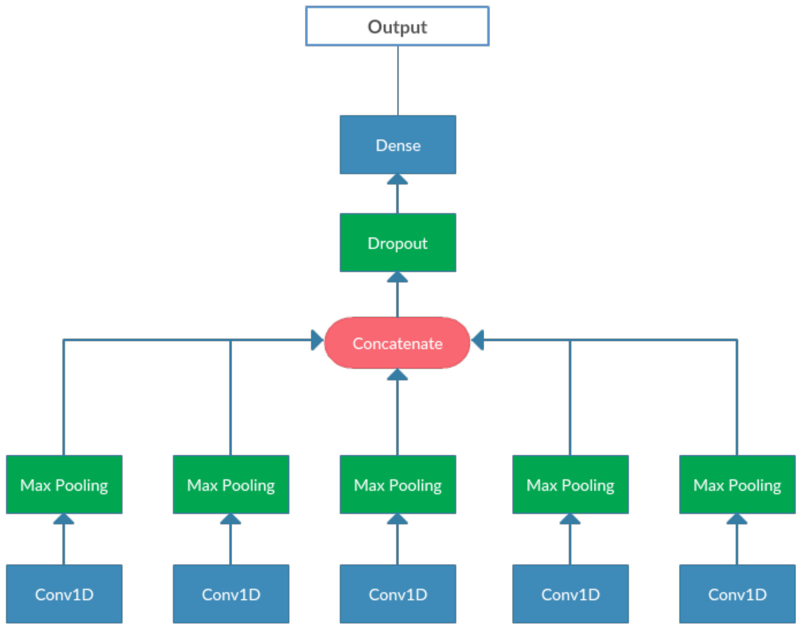

Fig. 4. CNN for case prediction classification

For classification using CNNs, we merged 5 convolutional 1D layers which have output dimension of 256 and filter length from 1 to 5. Lecun Uniform distribution was used to initialize the weights of layers. Input dimension for each layer was word2vec embedding's size, input length is set to a fixed number and 'tanh' is used as an activation function. To limit overfitting we also used kernel l1-l2 regularizer with $\lambda = 0.01$. Further, we added a max pooling layer to each of the Convolutional 1D layers. On the top of the merge, we added a dropout layer of 0.5 for further regularization. After this, we add a flattening layer. Finally, a dense layer is added to convert the output to proper dimensions. We used Keras library for building this Convolutional Network[2]. CNN was trained for 15 epochs using mini batch training and word2vec embedding generated from court cases as input.

5.2 Circuit to SCOTUS

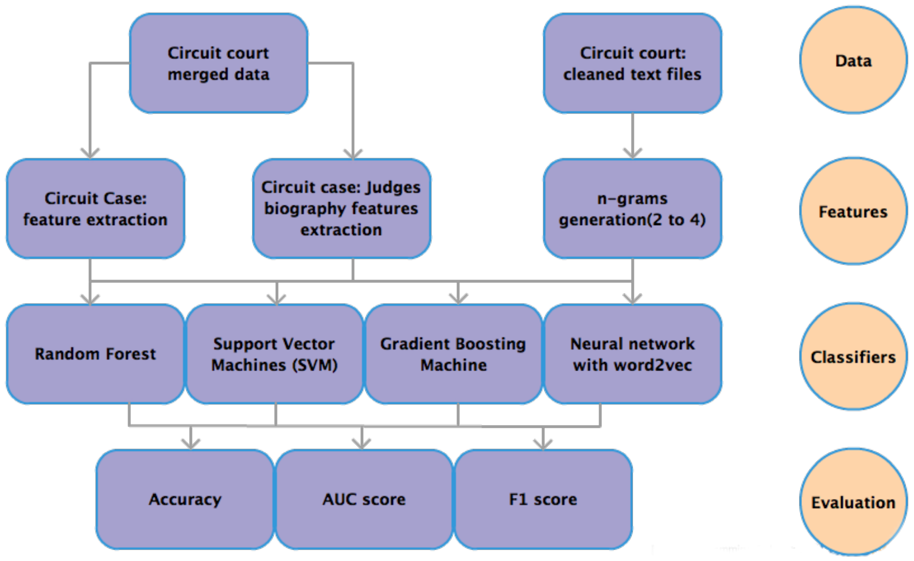
Circuit court merged data
Circuit court: cleaned text files
Data
Circuit Case: feature extraction
Circuit case: Judges biography features extraction
n-grams generation(2 to 4)
Features
Random Forest
Support Vector Machines (SVM)
Gradient Boosting Machine
Neural network with word2vec
Classifiers
Accuracy
AUC score
F1 score
Evaluation

Fig. 5. Pipeline diagram for circuit court to SCOTUS

Similar to the district-to-circuit-court once we had all the features ready, we used these features to train various classifiers, starting with linear SVM. Looking at results from linear SVM classifiers we concluded that, using n-grams as features its possible to predict the affirms/reversals better than random classifier. Then to improve our predictions we moved to ensemble based classifiers, we trained Random Forest and Gradient boosting with regression trees classifiers. Then instead of using n-grams as features we used Convolutional Neural Networks with Word2Vec embeddings. Using varied metrics to calculate the accuracy we were able to choose the classifier that performed best.

5.2.1 Training: Similarly we also had to first vectorize the n-grams before using them as input into sklearn classifiers. For this we used dict vectorizer provided by sklearn. Then these vectorized n-grams and label classes(Affirmed/Reversed) were used as input to various classifiers. While using word2vec model instead of using n-grams, we generated the word embeddings for the text and used this to train our basic convolutional neural network model.

6 Results

6.1 District to Circuit

For district to circuit we have the best results from the CNN trained for prediction. We used F1 measure as our prediction metric. We have tabulated our results for baseline classifiers and ensembles below:

Classifier	F1 Measure	Accuracy
Gradient Boosting (RT)	0.70	0.76
Multinomial NB	0.68	0.73
Random Forest Classifier	0.65	0.71
LinearSVC Classifier	0.65	0.68
Logistic Regression	0.67	0.70

Table 1. Results for baseline classifier

After observing the confusion matrix, we realized that the results weren't that descriptive of each class. Most of the results tend to align towards affirmed class. We overcame this situation in CNN. The graphs below show number of epochs versus various metrics for training and validation data.

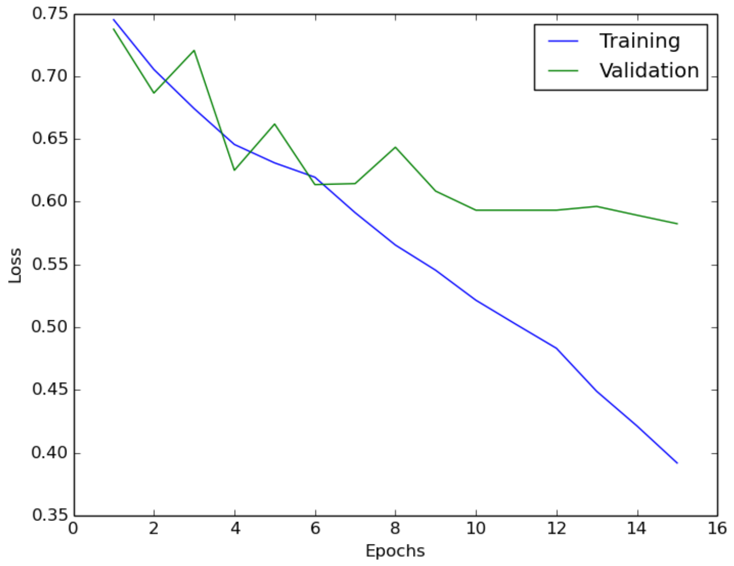

Fig. 6. Loss vs Number of epochs

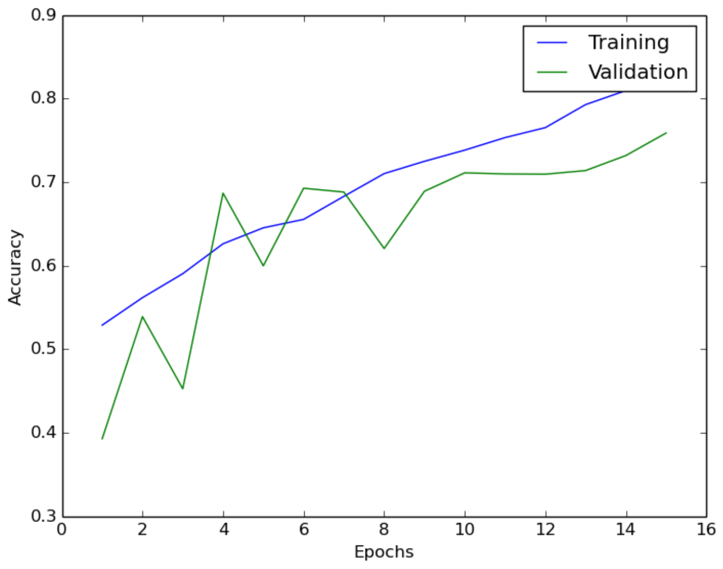

Fig. 7. Accuracy vs Number of epochs

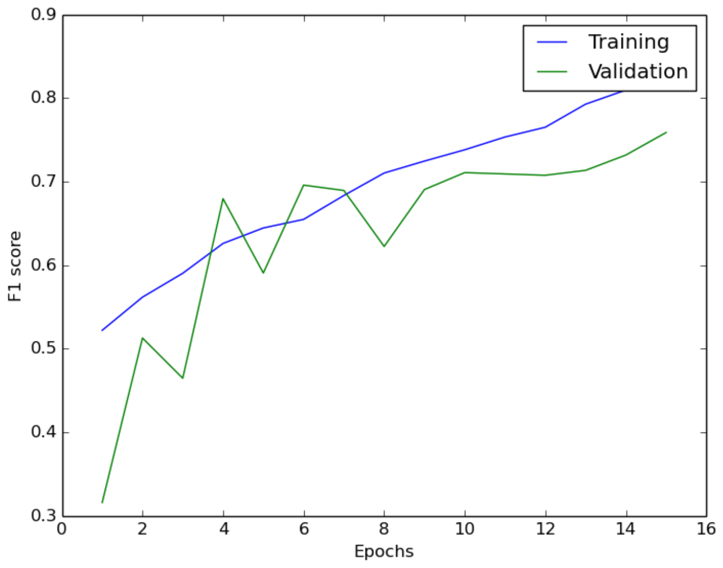

Fig. 8. Fmeasure vs Number of epochs

We achieved an accuracy of 0.79 and fmeasure of 0.75 using CNNs. The results still tend to align towards Affirmed cases but they are better than baseline models and ensembles.

We have also generated a word cloud of important words from the random forest classifier.

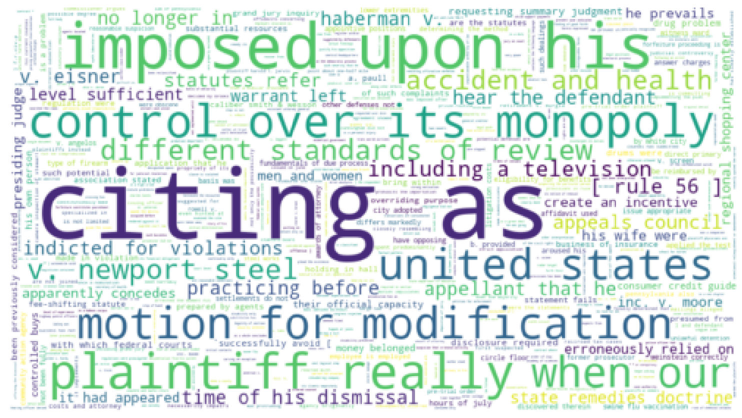

Fig. 9. Word cloud for district to circuit

1. We observe that in most of baseline models the ngrams fail to perform well. Even with the addition of judges information change in accuracy and f1 score is not significant.
2. CNN based model overfits if proper regularization is not used. With proper regularization, validation loss, accuracy and f1 measure improve with number of epochs.
3. Filtering out most of the ngrams based on $\frac{1}{8^{th}}$ strategy as mentioned above doesn't really bring a lot of change in accuracy and f1 score. This proves that only the top most occurring and top least occurring ngrams prove to be valuable features.
4. Adding the judges data doesn't improve the model by significant percentage. A probable cause for this might be that only a percentage the judges data was not matched perfectly.
5. Word cloud shows that "United States" as an appellee or appellant has a high significance in court decisions.
6. Some words which put significance on a statement like "really, control" have greater significance in decision.
7. A case that is properly cited has less chance of reversal.

6.2 Circuit to SCOTUS

Classifier	Accuracy	AUC	F1
Random Forest	63.5	0.54	0.77
Gradient Boosting	62.8	0.54	0.75
SVM	61.7	0.52	0.71
Word2Vec	61.2	0.51	0.81

Table 2. Various Classifier Metrics.

From our experiments we found that Random Forest gives us the best accuracy for the final data set. Though, the base Word2Vec model we implemented gave the best F1 score, the confusion matrix suggested that it was predicting reversed as the outcome for most cases.

Data Model	Accuracy	AUC
Case Text	63.5	0.54
Case Text + Judges Biography	66.9	0.55
Case Text + Judges Biography + Case Characteristics	68.1	0.56

Table 3. Random Forest Classfier.

We can see an evident improvement in prediction as we add more features into the Random forest Classifier.

7 Conclusion

7.1 District to Circuit

In this project, we tried to predict the reversal of a district court decision by a circuit court using the summarized decision text of the district court. We also considered the aspects of judges of the district court as a feature in our predictions. Since, the data was imbalanced, we used oversampling to reduce its effect. We observed that CNN performed the best among all classification techniques used. The results were attained by exhaustive experiments and trials with CNN and many other classification models. From the word cloud obtained, we found that some words in the decision text have a greater effect in the decision such as "United States" as an appelle or an appellant has high significance. We also discovered that some particular action verbs have more impact on the decision of the court. Features describing the judges (age, gender and locality) were found to be insignificant while predicting reversal of the district opinion.

7.2 Circuit to SCOTUS

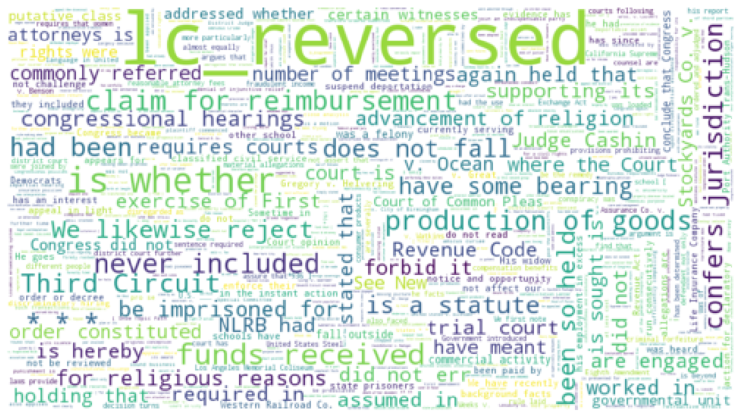

Fig. 10. Word Cloud showing Important Features

Examining the results we found that predicting the Supreme court reversals based on the text of the circuit court alone does not give great results. Adding additional features helped to improve the accuracy. From the word cloud we can see that the lower court disposition, i.e, whether the case was affirmed or reversed in the lower court was shown to have a great impact on the prediction scores. The word2vec model without proper regularization and tuning overfits the data as the inherent importance of memes generated through n-grams is not fully captured.

8 Future Work

8.1 District to Circuit

In the future, we can train our model with state of the art convolutional neural networks [9] to discover underlying relations which haven't been discovered. The word cloud depicts that citations are an important aspect in decision of a circuit court. Thus, after generating a proper citation graph for district cases, we can merge the citation level affirmed or reversal to our current model. Another important aspect would be to try out similar cases' information. We can create a cluster of similar cases using the information of judge, text and location. Then, we can add affirmed or reversed information of similar district court cases to our model. .

8.2 Circuit to SCOTUS

Citations is an important aspect which can be incorporated into the model. The citation graph will give the influence of previous cases on current decision and can identify weak precedents. Furthermore ,the Circuit of origin[7] of the case can also be factored into the prediction as historically it has been seen to have an effect. After identifying the important features their weights can be tuned more to give better results. The word2vec model showed much promise in predicting reversals from District to Circuit court cases. Hence, we feel that concentrating on improving its parameters in our basic model should give better results.
Our analysis totally excludes all information that is not available before the case is appealed in the supreme court. Hence the models can make better predictions if features such as the biography of supreme court judges are included. Adding data from the Supreme Court also opens the avenue for other predication such as reversal at the vote level of each of the nine judges. Further, this can be aggregated to predict the dissent between judges in the panel.

References

1. William B Cavnar, John M Trenkle, et al. N-gram-based text categorization. *Ann Arbor MI*, 48113(2):161–175, 1994.
2. François Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.
3. Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.
4. Yoon Kim. Convolutional neural networks for sentence classification. *EMNLP 2014*, 2014.
5. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
6. Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. `http://is.muni.cz/publication/884893/en`.
7. Stephen Wermiel. Scotus for law students (sponsored by bloomberg law): Scoring the circuits. `http://www.scotusblog.com/2014/06/scotus-for-law-students-sponsored-by-bloomberg-law-scoring-the-circuits/`, Jun. 22, 2014, 10:28 PM.
8. Jieping Ye, Ravi Janardan, Qi Li, and Haesun Park. Feature reduction via generalized uncorrelated linear discriminant analysis. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1312–1322, 2006.
9. Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.