Legal Judgement Prediction for UK Courts

Benjamin Strickson
University of East Anglia
Norwich, UK
benjamin.strickson@gmail.com

Beatriz De La Iglesia
University of East Anglia
Norwich, UK
B.Iglesia@uea.ac.uk

ABSTRACT
Legal Judgement Prediction (LJP) is the task of automatically predicting the outcome of a court case given only the case document. During the last five years researchers have successfully attempted this task for the supreme courts of three jurisdictions: the European Union, France, and China. Motivation includes the many real world applications including: a prediction system that can be used at the judgement drafting stage, and the identification of the most important words and phrases within a judgement. The aim of our research was to build, for the first time, an LJP model for UK court cases. This required the creation of a labelled data set of UK court judgements and the subsequent application of machine learning models. We evaluated different feature representations and different algorithms.

Our best performing model achieved: 69.05% accuracy and 69.02 F1 score. We demonstrate that LJP is a promising area of further research for UK courts by achieving high model performance and the ability to easily extract useful features.

CCS Concepts
• Information systems→Content analysis and feature selection
• Information systems→Clustering and classification
• Information systems→Document topic models.

Keywords
Legal judgement prediction; legal calculus; feature extraction.

1. INTRODUCTION
The ability of computers to predict the outcome of legal cases from the text documentation began to attract serious attention from the 1960s onward. Lawlor [13] argued that it should be possible to predict how the facts and legal arguments of a case would be received by a judge. In the subsequent decades several studies [2, 3, 26] have attempted to manually derive a legal calculus, which is defined as an abstract system of argument structures or schemes that are linguistically realised in legal texts.

A new approach [21], based on machine learning techniques, emerged to move the field forward. The first attempts [9, 11, 17] used machine learning models to predict the judgements of the Supreme Court of the United States (SCOTUS). These attempts used document tags as predictive features, tags such as type of case and judge name.

ICISS 2020, March 19–22, 2020, Cambridge, United Kingdom
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-7725-6/20/03…$15.00
https://doi.org/10.1145/3388176.3388183

Subsequent researchers built predictive models which relied only on unstructured text features, this became known as LJP. Aletras et al. [1] were the first to apply this approach, attempting LJP on the European Court of Human Rights (ECHR) data set. Three subsequent published studies share a common methodology and research objectives, two of those also used the ECHR data set [16, 18], and the other used the French Court of Cassation data set [19]. Our research also aims to rely solely on text-derived features and machine learning. It will be the first study to attempt LJP for UK court decisions and represents further important evidence of the potential to successfully apply machine learning for LJP.

There are two significant problems this study is required to overcome when attempting LJP on UK court documents. The first is the currently limited ability of Natural Language Processing (NLP) techniques to recognise complex semantic structures such as arguments. The second problem is specific to the UK; there is currently no structured public data set of UK court cases.

Our research aim is to build an interpretable predictive model for UK court cases using only the court documents. The objectives that will help us to achieve this aim are:

- To build a labelled data set of UK court judgements with an outcome variable that can be used in prediction tasks.
- To build a prediction model using machine learning techniques previously applied by comparable studies.
- To test alternative text mining techniques such as word embedding features with neural network models.

In this paper we explain how we built our labelled data set for UK court judgements and then used it to test existing and newer text mining techniques obtaining good accuracy and usable features. Section 2 reviews similar work done by other researchers; Section 3 explains our methodology; Sections 4 and 5 present our text mining results and discussions; and finally we present our conclusions in Section 6.

2. RELATED WORK
One of the very important decisions that affects any text mining application is how to represent text as features that can be handled by machine learning algorithms. The n-gram has become a hugely popular feature set in text classification tasks, where a gram is often equivalent to a word. First utilised by Shannon [23], its main advantage is that it allows documents to be represented as vectors. All individual words (one-grams) or larger n-grams from the corpus are represented as columns, and each document is represented as a single row in the matrix. The value at the intersection of the row and column represents how often a term, or n-gram, appears in a document. This value is most commonly a simple count statistic or the Term Frequency-Inverse Document Frequency (TFIDF) [25] statistic. These vector space models proved beneficial for the application of machine learning models.

The first paper to apply these features to the task of LJP was Aletras et al. [1], who used n-grams ranging from one-grams to four-grams. Three other published studies [16, 18, 19] followed using the same methodology.

An alternative approach known as generative language models has also been applied to extract feature sets for the task of text classification. Blei et al. [5] developed the Latent Dirichlet Allocation (LDA) algorithm which implements this theory by constructing topic clusters out of text documents. The algorithm assumes that documents are composed of a random mixture of latent topics, and each of the topics is characterised by a distribution over words. Some studies [5, 15] have been able to demonstrate a performance benefit in text classification tasks to support the application of LDA feature sets.

A recent alternative to the vector space feature set emerged to address their short-comings; Bengio et al. [4] argued that n-gram feature sets were problematic as they did not consider the similarity between words. They developed a technique known as neural network-based word embeddings in which each word is represented as a vector with multiple dimensions. These dimensions contain values that encode information concerning the target word's surrounding words. This work was supported by Mikolov et al. [20] who developed the Word2Vec algorithm that we will use. This algorithm has two distinct phases: the first phase involves training a neural network to learn word distributions. We will use the Continuous Bag Of Words (CBOW) architecture, where a window of surrounding words is used to predict a target word. The second phase feeds the word vectors from the learned hidden layer into an output layer, to represent each word as an n-dimensional vector.

In terms of classification algorithms, we looked at the current LJP literature to select suitable models. All the LJP studies mentioned so far have used Support Vector Machine (SVM)s. Their popularity is due to their high performance across a range of text classification tasks [10, 27, 29]. Additionally we included the Logistic Regression (LR), Random Forest (RF), and k-Nearest Neighbour (k-NN) as used by Liu and Chen [16]. To meet our third research objective we included two neural networks: a Single Layer Perceptron (SLP) and a Multi-Layer Perceptron (MLP).

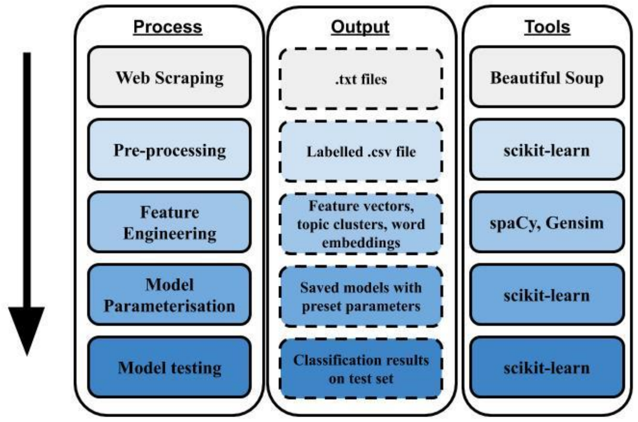

Figure 1. Research methodology workflow.

3. RESEARCH METHODOLOGY

3.1 Data Collection

The data set used in this study was restricted to the judgements issued by the UK's highest court of appeal, similar to other studies [1, 6, 16, 18, 19] which focused on one court within one country's legal system. To collect these judgements a web scraper was built.

3.2 Data Labelling

Each web-scraped case file was divided into the separate judgements passed down by the individual judges. Next, each of these rulings were automatically labelled into 'allow' or 'dismiss' using a pattern matching approach. The files which could not be labelled with an outcome were individually examined before being excluded from the data set. They represent the cases where no final judgement was reached by the judges. To check the accuracy of the labelling methodology a random stratified sample of 5% of the data set was examined. A total of 4,959 text files were labelled after exclusions. This is comparable to three other studies where 584 cases [1, 16] and 3,132 cases [18] were used. It is, however, far fewer than the number of cases used in the study by Medvedeva et al. [19] on the French court of Cassation which had 126,865 cases. Our data collection methodology and code has been made publicly available[1].

3.3 Pre-processing

Text that identified with the outcome labels was removed from the data set to avoid giving the classifier the obvious information; this approach is inline with three other studies [1, 18, 19]. However, Liu and Chen [16] do not mention this stage in their review, we are thus cautious about their model results. To ensure that no words could be used as proxies for the labels an additional review of the most highly correlated model features was performed.

The remaining text consisted of a total set of 188,294 unique tokens (words). Reducing this high degree of dimensionality in our data set was considered important to prevent generalisation error and model over fitting. Therefore, we used the pre-processing steps set out by Joachims [10] as our guide and we achieved a significant reduction. The results of applying the different pre-processing steps such as converting to lowercase, removing numbers and stop words and lemmatization are presented in Table 1.

Table 1. Pre-processing steps and resulting token count

	Unique token (word) count
No pre-processing	188,294
Lowercase conversion	170,126
Numbers removed	166,949
Stop words removed	166,638
Lemmatize	157,648

3.4 Feature Engineering

To investigate which features gave best results for our specific environment, our feature sets were: n-grams, topic clusters and word embeddings. For n-grams we used the standard count and TFIDF implementations. We set n as one of the parameters to be optimised with a range from one to four. For the topic clusters we decided upon the LDA algorithm; a popular topic modelling

[1] Code available at https://github.com/BStricks/legal_document_classifier_V2

algorithm [5, 15]. We set the number of topics as one of the parameters to be optimised, ranging from five to thirty. For the word embedding feature set there was only one relevant study to draw from [6]. We chose to use untrained embeddings because our corpus contained a large number of words that was unique to the legal domain. These context specific words could have been problematic for pre-trained models such as those trained on Google's news feed. We also chose to use the Doc2Vec algorithm [14], which is an extension of the original Word2Vec algorithm that is able to generate document level vectors.

3.5 Model Tuning

All the above algorithms were implemented in the Python 3 language using the Scikit-learn package [22]. Below are the optimal parameters for each algorithm found by cross-validated random search, taken from the feature set that performed best:

SVM and TFIDF vectors: n-gram range (1,3), minimum feature occurrence (1), maximum features (10000), kernel (linear), c (5).

RF and TFIDF vectors: n-gram range (1,4), minimum feature occurrence (4), maximum features (4000), number of estimators (1000), max depth (20).

LR and TFIDF vectors: n-gram range (1,2), minimum feature occurrence (4), maximum features (None), solver (lbfgs), c (5).

k-NN and Doc2Vec: clusters (5).

SLP and TFIDF vectors: n-gram range (1,2), minimum feature occurrence (2), maximum features (10000).

MLP and TFIDF vectors: n-gram range (1,3), minimum feature occurrence (4), maximum features (10000), solver (adam), hidden layers (2,2), activation (logistic).

We report also for comparison the accuracy of Scikit-learn's dummy classifier which respects the training set's class distribution [22].

3.6 Evaluation

Our data set was split into two partitions. The first 80% of the data was used for a ten-fold cross-validated random search for hyperparameter optimisation with Scikit-learn [22]. The final 20% of the data, was used as a test set to report model scores. In LJP research average accuracy is the most commonly reported model score, we will report this for our test data. Our study will additionally report: F1, precision, and recall, as they provide important additional information on performance.

4. RESULTS

As shown in Table 3, the top performing combination of model and feature set was the LR algorithm paired with the TFIDF vector representation. This combination achieved an F1 score of 69.02, a precision of 69.05% and a recall of 69.02%. Overall both the RF and LR algorithms performed well across feature sets. SVM, k-NN, and SLP algorithms tended to perform worse across most of the feature sets. Overall the best performing feature sets were the Count and TFIDF vectors, with the topic clusters and word embeddings feature sets performing worse.

For the majority of the model and feature set pairings the F1, recall, and precision scores were roughly equivalent. This is partly due to having a balanced data set; with 2,525 'Allow' cases and 2,434 'Reject' cases. It also suggests that the models are generally good at selecting true positives and avoiding false positives, as well as selecting many of the relevant data points, avoiding false negatives. Additionally the best performing models from each feature set models were analysed with Receiver Operating Characteristic (ROC) curves. This was done to better understand performance at various threshold levels of sensitivity (True Positive Rate or recall) and specificity (or False Positive Rate). The curves, see Fig. 2, support our initial observations of stronger model performance for the count and TFIDF vector feature sets.

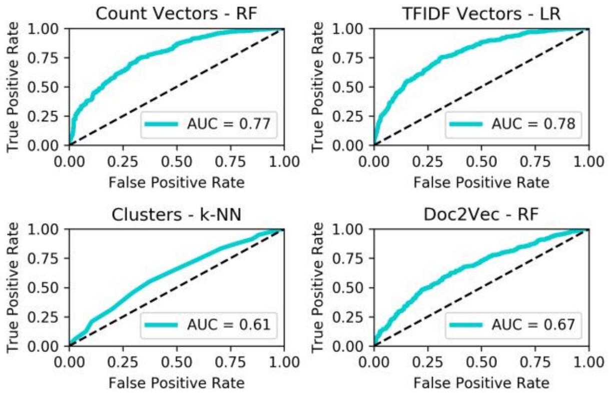

Figure 2. ROC curves for best model and features pairings.

5. DISCUSSION

5.1 n-grams

The strongest performing feature set as measured by the prediction scores was the TFIDF vectors composed of one-grams to two-grams. Given other text classification study results in this area, the high performance of this feature set was expected. Next we considered the usability of the n-gram features, this was an important secondary consideration when evaluating model performance. We extracted the most important n-gram features for our strongest performing vector-space model. Separating the features by outcome label, these are presented in Table 2. A preliminary review shows that these features contain interpretable meanings, and that most of them would generalise well to new cases.

Table 2. Most important n-gram features for the LR model extracted from the TFIDF representation associated with each outcome

	Top 15 most important features
Reject	rely, Iraq, instance, submit, minimum, actual, main, wreck, hire, hall, covenant, territory, regime, agency, product
Allow	restore, remit, siac, cross appeal, restore order, carrier, situation, segregation, account, declaration, directive, commission, avoid, perform, long

Table 3. Results showing Accuracy, F1, Precision and Recall measurements on the test data for different model and feature pairings

	Count vectors				TFIDF vectors				LDA topic clusters				Word embeddings			
	Acc.	F1	Prec.	Rec.	Acc.	F1	Prec.	Rec.	Acc.	F1	Prec.	Rec.	Acc.	F1	Prec.	Rec.
Dummy	52.02	52.01	52.02	52.02	49.09	49.04	49.05	49.05	49.90	49.90	49.91	49.91	49.60	49.59	49.64	49.64
SVM	61.49	61.49	61.50	61.50	65.93	65.89	65.92	65.89	52.12	50.47	52.05	51.80	60.79	60.11	62.58	61.45
RF	**66.13**	**66.12**	**66.12**	**66.13**	66.63	66.62	66.62	66.61	56.25	55.93	56.26	56.11	**64.21**	**64.17**	**64.17**	**64.18**
LR	65.12	65.12	65.13	65.13	**69.05**	**69.02**	**69.05**	**69.02**	53.12	53.11	53.11	53.11	62.10	62.00	62.66	62.42
k-NN	61.79	61.76	61.93	61.87	59.98	59.86	60.31	60.11	**57.76**	**57.63**	**58.01**	**57.88**	62.70	62.59	63.33	63.05
SLP	62.50	62.50	62.52	62.52	64.72	64.71	64.79	64.76	50.91	33.73	25.45	50.00	57.56	53.00	59.55	56.34
MLP	65.62	65.85	65.62	65.58	66.94	66.94	66.95	66.95	55.95	55.71	55.93	55.83	61.59	61.58	61.61	61.63

5.2 Topic Clusters

Topic clusters performed relatively poorly compared to the other feature sets, with the best performing model achieving an F1 score of 57.63. Despite these low scores, we assessed their usability by extracting the main features for each topic in Table 4. We can see promising results with each of the topics appearing to coalesce around similar legal areas and terminology. However, a full review of the topics by a trained lawyer may be necessary, though it is considered outside the scope of this paper.

Table 4. Topic clusters and the most important features associated with each. ('pron' is code for all pronouns)

Topic number	Top 5 most important topic features
1	act, pron, provision, section, parliament
2	pron, court, right, tax, company
3	pron, order, court, make, sentence
4	pron, act, lord, rule, board
5	pron, lord, friend, noble, pron noble
6	pron, company, pay, tax, payment
7	pron, article, right, state, convention
8	pron, offence, criminal, act, police
9	pron, child, case, pron, pron court
10	pron, court, decision, case, appeal
11	pron, law, state, court, jurisdiction
12	pron, regulation, work, member, directive
13	pron, case, claim, damage, lord
14	pron, right, property, land, use
15	pron, contract, party, agreement, clause

5.3 Word Embeddings

Our experiments show that the Doc2Vec word embedding feature set performed reasonably well with the best model achieving an F1 score of 64.17. The expectation had been that word embeddings would deliver the best overall model, given that other researchers have used this feature set to achieve state of the art results. Our explanation for the observed results is that whilst the word embeddings incorporated more contextual information than the other feature sets, the low number of data points in our data set may not have provided the necessary context. Given the success found elsewhere with pre-trained word embeddings we could attempt to construct these for future studies. A potential corpus constructed of all legal judgements from UK courts would provide ideal pretraining for our task. Finally we considered feature usability; it is noted that they provide significantly less insight than our alternative feature sets.

5.4 Machine Learning Algorithms

We can say that the k-NN and RF algorithms delivered the most consistent results across the feature sets. Whereas the SVM and LR algorithms less consistent performance, determined partly by feature set. We can say that in almost all cases the machine learning algorithms performed better with a reduced feature space. As demonstrated by the parameters selected during cross-validation.

5.5 Neural Nets

Our choice of artificial neural network architectures for our LJP task did not provide a clear improvement over our standard machine learning models. We used two of the simplest models, whereas text mining researchers working in other domains have recently applied more complex architectures with good results. As expected the MLP out-performed the SLP; this was most likely due to the complex nature of the classification problem. MLP's have an ability to handle complex decision boundaries which may exist in this problem.

Convolution Neural Networks [12, 30], Recurrent Neural Networks [24] or other Deep Learning algorithms may be more suitable for word embeddings [7, 8]. Also, the unpublished results of Chalkidis et al. [6] show the promise of the Hierarchical Attention Network (HAN) architecture. Their models achieved state of the art performance, due to the HAN architecture's suitability for document classification [28].

It is also worth considering that given the black box nature of these algorithms it is much harder to understand which features are used by the models when making their predictions. This means that even those studies [6] which have successfully used neural networks to boost LJP results have been unable to justify model decisions. Current research [6] indicates that there is the potential for attention scores to be used as a proxy for feature extraction; however, this metric has not been thoroughly reviewed or tested for use by legal experts.

6. CONCLUSION AND FURTHER WORK

Against our first objective we achieved the creation of 100 years of labelled UK court judgements. Against our second objective we delivered a predictive model that was both accurate and highly interpretable. We found that both vector space feature sets were able to deliver good results, though TFIDF features paired with the LR algorithm achieved the highest F1 score of 69.02. Extracting the most important features from the vector space and topic cluster models was a relatively easy task and indicates good potential model usability. Our third objective was the application of word embeddings and neural networks to the task of LJP. Our results were unable to show that word embeddings combined with our choice of neural networks could improve model performance.

A number of more advanced neural network architectures have been used to great effect in other text classification tasks, we believe these could be used to great effect for LJP. Significantly improving the results we had with the SLP and MLP algorithms. The use of ngrams and topic clusters proved successful as predictive feature sets, however, in order to understand their usefulness further testing is needed. Our proposal would be for the independent examination and testing of the extracted features by professional lawyers.

7. REFERENCES

[1] Nikolaos Aletras, Dimitrios Tsarapatsanis, Daniel Preoţiuc-Pietro, and Vasileios Lampos. 2016. Predicting judicial decisions of the European Court of Human Rights: A natural language processing perspective. *PeerJ Computer Science* 2 (2016), e93.

[2] Kevin D. Ashley. 1991. *Modeling Legal Arguments: Reasoning with Cases and Hypotheticals*. MIT Press, Cambridge, MA, USA.

[3] Trevor Bench-Capon. 1997. Argument in artificial intelligence and law. *Artificial Intelligence and Law* 5, 4 (1997), 249–261.

[4] Yoshua Bengio, R ǵean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research* 3, Feb (2003), 1137–1155.

[5] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[6] Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019. Neural Legal Judgment Prediction in English. *CoRR* abs/1906.02059 (2019). arXiv:1906.02059 http://arxiv.org/abs/1906.02059

[7] Oduwa Edo-Osagie, Beatriz de la Iglesia, Iain R. Lake, and Obaghe Edeghere. 2019. Deep Learning for Relevance Filtering in Syndromic Surveillance: A Case Study in Asthma/Difficulty Breathing. In *ICPRAM*.

[8] Oduwa Edo-Osagie, Iain Lake, Obaghe Edeghere, and Beatriz DeÂăLa Iglesia. 2019. Attention-Based Recurrent Neural Networks (RNNs) for Short Text Classification: An Application in Public Health Monitoring. In *Advances in Computational Intelligence*, Ignacio Rojas, Gonzalo Joya, and Andreu Catala (Eds.). Springer International Publishing, Cham, 895–911.

[9] Roger Guimer àand Marta Sales-Pardo. 2011. Justice blocks and predictability of us supreme court votes. *PloS one* 6, 11 (2011), e27188.

[10] Thorsten Joachims. 1998. Text categorization with Support Vector Machines: Learning with many relevant features. In *Machine Learning: ECML-98,* Claire N édellec and C éline Rouveirol (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 137–142.

[11] Daniel Martin Katz, Michael J Bommarito II, and Josh Blackman. 2017. A general approach for predicting the behavior of the Supreme Court of the United States. *PloS one* 12, 4 (2017), e0174698.

[12] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent Convolutional Neural Networks for Text Classification. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15)*. AAAI Press, 2267–2273. http://dl.acm.org/citation.cfm?id=2886521.2886636

[13] Reed C Lawlor. 1963. What computers can do: Analysis and prediction of judicial decisions. *ABAJ* 49 (1963), 337.

[14] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32 (ICML'14)*. JMLR.org, II–1188–II–1196. http://dl.acm.org/citation.cfm?id=3044805.3045025

[15] Sangno Lee, Jaeki Song, and Yongjin Kim. 2010. An empirical comparison of four text mining methods. *Journal of Computer Information Systems* 51, 1 (2010), 1–10.

[16] Zhenyu Liu and Huanhuan Chen. 2017. A predictive performance comparison of machine learning models for judicial cases. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, IEEE, 1–6.

[17] Andrew D Martin, Kevin M Quinn, Theodore W Ruger, and Pauline T Kim. 2004. Competing approaches to predicting supreme court decision making. *Perspectives on Politics* 2, 4 (2004), 761–767.

[18] Masha Medvedeva, Michel Vols, and Martijn Wieling. 2018. Judicial Decisions of the European Court of Human Rights: Looking into the Crystal Ball. In *Proceedings of the Conference on Empirical Legal Studies*.

[19] Masha Medvedeva, Michel Vols, and Martijn Wieling. 2019. Using machine learning to predict decisions of the European Court of Human Rights. *Artificial Intelligence and Law* (26 Jun 2019), 1–30. https://doi.org/10.1007/s10506-019-09255-y

[20] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Atlanta, Georgia, 746–751. https://www.aclweb.org/anthology/N13-1090

[21] Marie-Francine Moens, Erik Boiy, Raquel Mochales Palau, and Chris Reed. 2007. Automatic Detection of Arguments in Legal Texts. In *Proceedings of the 11th International Conference on Artificial Intelligence and Law (ICAIL '07)*. ACM, New York, NY, USA, 225–230. https://doi.org/10.1145/1276318.1276362

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[23] Claude Elwood Shannon. 1948. A mathematical theory of communication. *Bell system technical journal* 27, 3 (1948), 379–423.

[24] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Seattle, Washington, USA, 1631–1642. https://www.aclweb.org/anthology/D13-1170

[25] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* 28, 1 (1972), 11–21.

[26] Adam Wyner and Trevor Bench-Capon. 2007. Argument Schemes for Legal Case-based Reasoning. In *Proceedings of the 2007 Conference on Legal Knowledge and Information Systems: JURIX 2007: The Twentieth Annual Conference*. IOS Press, Amsterdam, The Netherlands, The Netherlands, 139–149. http://dl.acm.org/citation.cfm?id=1565610.1565629

[27] Yiming Yang and Xin Liu. 1999. A Re-examination of Text Categorization Methods. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '99)*. ACM, New York, NY, USA,42–49. https://doi.org/10.1145/312624.312647

[28] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical Attention Networks for Document Classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California, 1480–1489. https://doi.org/10.18653/v1/N16-1174

[29] Tong Zhang and Frank J Oles. 2001. Text categorization based on regularized linear classification methods. *Information retrieval* 4, 1 (2001), 5–31.

[30] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'15)*. MIT Press, Cambridge, MA, USA, 649–657. http://dl.acm.org/citation.cfm?id=2969239.2969312