

LAW IDENTIFICATION AND CLASSIFICATION TEXT BOT USING MODIFIED ELBOW METHOD AND BIDIRECTIONAL ENCODER REPRESENTATION FROM TRANSFORMERS

Jain Kabir¹, Shah Mahir Hitesh¹, Lee Theophilus Khong Yoon¹

Mentor: Lim How Khang²

¹NUS High School of Mathematics and Science, 20 Clementi Ave 1, Singapore 129957

²Singapore Management University, Yong Pung How School of Law, 55 Armenian Street, Singapore 179943

August 2022

ABSTRACT

In our research, we aim to develop a system which can take the background information of a law case as input, and as output, return the related acts, relevant previous court proceedings and section of law it is related to. We employed and compared various Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al.) [1] models to vectorize the text which was later classified based on what laws it was related to. We clustered similar background cases to get related laws. We then implemented a novel modified elbow method, to ensure that the system would return the most relevant laws with the highest accuracy. We were able to implement this system that would be able to help both lawyers and common citizens obtain the relevant acts and details about their situation in a quick and efficient manner. We implemented this solution on telegram to make it more accessible to users. The current work done is focused on the context of Singapore, however this technique can be extended to the laws of other countries.

Academic Integrity

Research Report

2022 S.T. Yau High School Science Award (Asia)

Commitments on Academic Honesty and Integrity

We hereby declare that we

1. are fully committed to the principle of honesty, integrity and fair play throughout the competition.
2. actually perform the research work ourselves and thus truly understand the content of the work.
3. observe the common standard of academic integrity adopted by most journals and degree theses.
4. have declared all the assistance and contribution we have received from any personnel, agency, institution, etc. for the research work.
5. undertake to avoid getting in touch with assessment panel members in a way that may lead to direct or indirect conflict of interest.
6. undertake to avoid any interaction with assessment panel members that would undermine the neutrality of the panel member and fairness of the assessment process.
7. observe the safety regulations of the laboratory(ies) where the we conduct the experiment(s), if applicable.
8. observe all rules and regulations of the competition.
9. agree that the decision of YHSA(Asia) is final in all matters related to the competition.


We understand and agree that failure to honour the above commitments may lead to disqualification from the competition and/or removal of reward, if applicable; that any unethical deeds, if found, will be disclosed to the school principal of team member(s) and relevant parties if deemed necessary; and that the decision of YHSA(Asia) is final and no appeal will be accepted.

(Signatures of full team below)


Name of team member: Kabir Jain


Name of team member: Theophilus Lee


Name of team member: Alan Blon Mahin Anteha


Name of supervising teacher: Mr Lim Teck Chuan

Noted and endorsed by

(signature) 

MR GOH HOCK LEONG
Deputy Principal (Academic)
S.T. Yau High School of Math & Science

Name of school principal:

INTRODUCTION

Motivation

Law is very to the public. People often do not have the time and/or the knowledge to conduct legal research by themselves. Consultation of lawyers may prove too expensive. It usually costs more than \$100 per hour to get a commercial lawyer in Singapore alone.

Hence, we plan to utilize AI to open an avenue for free, accessible law help. The issue at hand is that many people find themselves inaccessible in legal predicaments where they would want to research about the laws affecting them but would not want to read through all penal codes and laws. Similarly, this tool could also benefit lawyers to find similar cases with similar precedents.

Legal System

In Singapore, there exists a judiciary system that governs what acts are illegal or not. Singapore's laws are organized under the penal code, which serves as the base document for Singapore's laws, and acts that further extend on or overwrite sections of the penal code. The penal code and acts are well organized and consistently structured, however, the legal terms and language used are not understandable by people who are not educated in law. This creates an issue for such people who want to conduct research themselves. This is the problem we wish to address.

We have partnered with Singapore's Community Justice Center to obtain data involving citizens requesting for legal assistance.

Our Solution

We plan to create a system that can take in the legal predicament and background information from the user's case. We would then be giving them section of law that the issue is covered under, previous cases and court proceedings that is similar to the current case, and finally the relevant acts that are related to the user's case. This will all be packaged into a Telegram bot, using the Telegram API [2], making it a lot more accessible to users.

COMPUTATIONAL METHODS AND RESULTS

Data Obtaining and Cleaning

Background Case Information

Background Information on the Singapore cases from 2016 to 2020 was obtained from Community Justice Centre. The data used consists of background information on cases categorized by 47 case types, stored in a CSV format. We made the observation that the dataset size for multiple case types was insufficient for accurate BERT classification. Hence, the data was reorganized into 12 case types, with 1 case type used for cases with miscellaneous type. The data was augmented to make it less bias and more evenly distributed between classes. Headers in background information were removed through regex filtering. Stop-words, as defined by the NLTK (Natural Language Toolkit) library [3], were then removed. NLTK library provides a comprehensive toolset of functions and resources commonly used in natural language processing. The NLTK stop-word list consists of general stop-

LAW IDENTIFICATION AND CLASSIFICATION TEXT BOT USING MODIFIED
ELBOW METHOD AND BIDIRECTIONAL ENCODER REPRESENTATION FROM TRANSFORMERS

words that are often encountered in daily English language. We have chosen NLTK’s stop-word list over more exhaustive listings of stop-words, as legal documents are often very precise in wording, hence giving many normally unimportant words significant meaning. Hence, NLTK was chosen as the more accurate list.

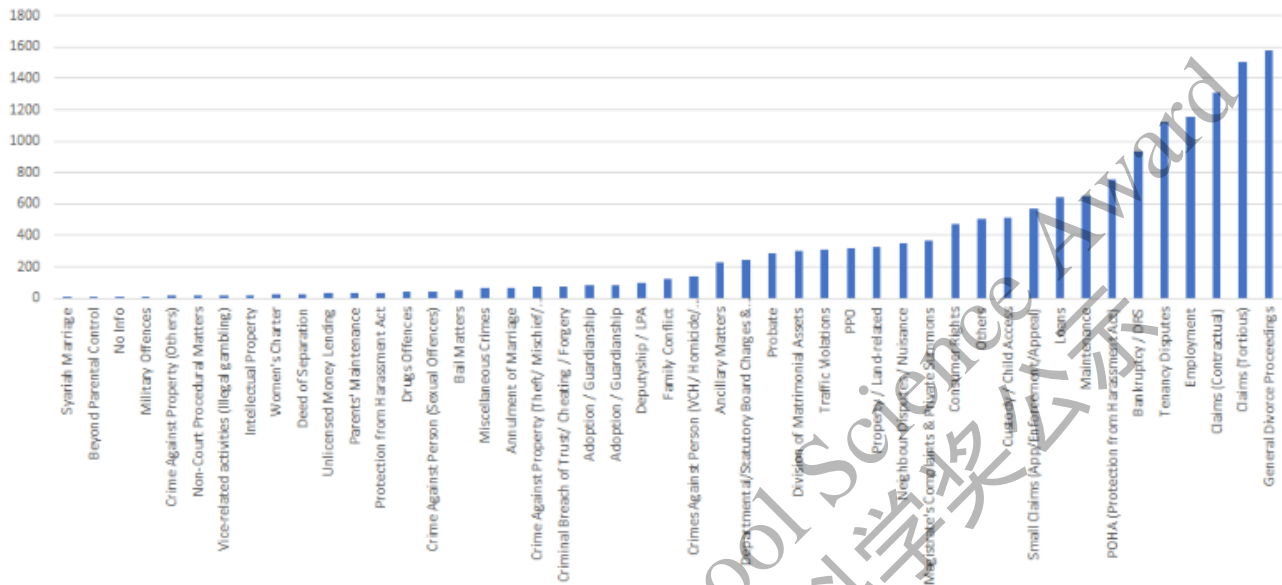


Figure 1: The raw data from the CTC

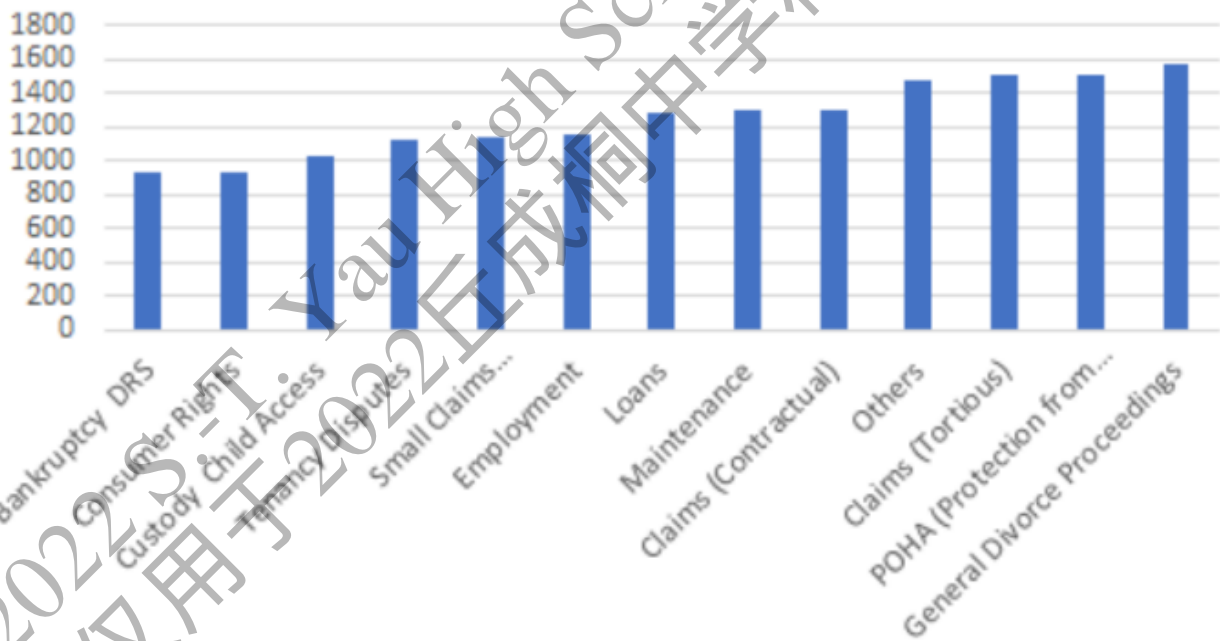


Figure 2: Data after augmentation and reduction

Supreme Court Case Information

Supreme Court Decisions for Singapore’s court cases from 2018 to 2022 were obtained from Commonlii [4]. The data used consists of PDF documents detailing court proceedings and judge’s decisions. Documents were first parsed to a plaintext format with python textract. Analysis of the documents surfaced that the tokens “Judgement Reserved” and “Introduction” are commonly used

before the Introduction section. Additionally, information before these tokens consists of glossaries, header metadata, and other content that is not useful for our system. Hence, the positions of these tokens were obtained using regex, and glossaries and header metadata were then removed from the document. An evaluation of the frequency of words was performed. We found that the most commonly occurring words are within the NLTK library stop-words list. Hence, stop-words, as defined by the NLTK library, were then removed. To extract act labels from documents, a list of all acts was required. An index of all Acts effective on 26 July 2022 was web scraped from Singapore Statutes Online[5]. A series of http GET requests were sent to Singapore Statutes Online, with PageSize=500 as argument, and appending '/X' to the path, where X is the page to obtain. This allows us to obtain the largest possible subset of the Act index per query. All other parameters were set to default. We then traverse each html document to obtain all <a> tags within the table of acts. Text content is extracted from the <a> tags as an index of all Act names. We made the observation that naming convention of acts within Supreme Court Decisions, is generally consistent with that of Singapore Statutes Online. Hence, logic to detect variations in act name formatting are not required. The court decision documents were then searched with regex for mentions of acts. The resulting labels were then delimited with newlines, and saved to an external text document for each court decision document.

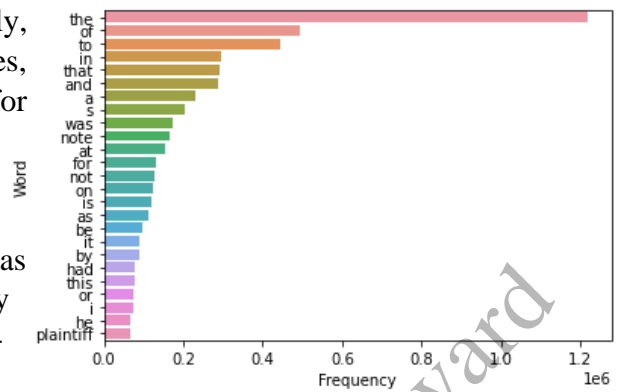


Figure 3: Graph of word frequency in Court Decisions

BERT Model and Classifier Head

BERT makes use of Transformer [6], an attention mechanism that learns contextual relations between words (or sub-words) in a text. As opposed to directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. Therefore, it is considered bidirectional, though it would be more accurate to say that it's non-directional. This characteristic allows the model to learn the context of a word based on all its surroundings (left and right of the word).

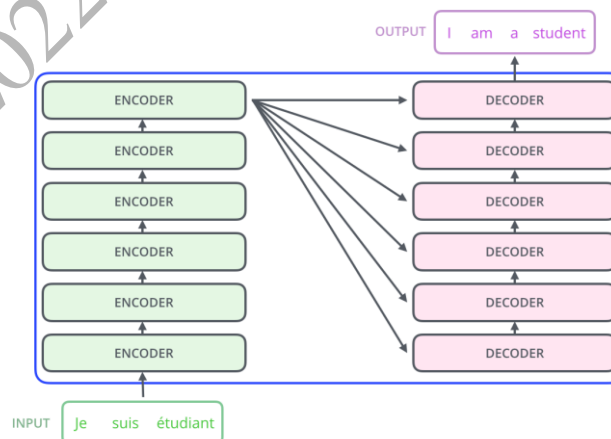


Figure 4: An simplified diagram of BERT

A classifier head is then added on the BERT model to be able to classify the text inputted.

The text inputs are first transformed to numeric token ids and are arranged in several Tensors before being input into the BERT. Then, the BERT encoder is used to pass the text into the BERT model itself. The Dropout layer [7] randomly sets input units to 0 with a frequency of rate at each step during training time, which helps prevent overfitting. Inputs not set to 0 are scaled up by $\frac{1}{1-rate}$ such that the sum over all inputs is unchanged. A dense layer for the classifier was finally added. This is where the text is classified.

This classifier was used to classify the segment of law the user's case is in, we will also be using it to classify the specific act the user's case is related to. The BERT models have been previously pretrained for alternative tasks, and the models are fine-tuned via transfer learning.

Transfer learning is a well-established technique for improving the performance of machine learning models, and previous papers [8] establishes that transformers are no different. Fine tuning transformers trained on rich datasets for downstream tasks achieves state of the art results for dozens of language understanding tasks. Due to their high performance in related language modelling tasks, these three models were utilized and train them using transfer learning.

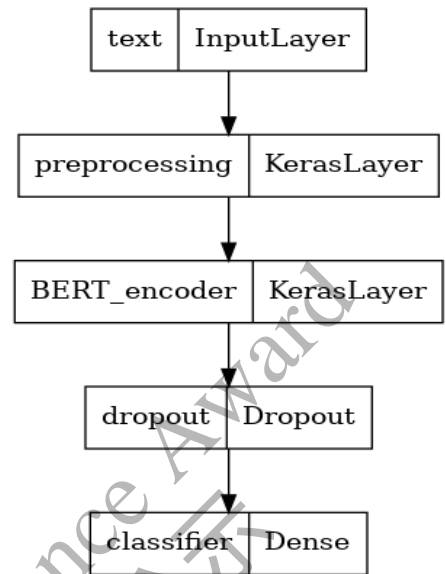


Figure 5: Diagram of the BERT

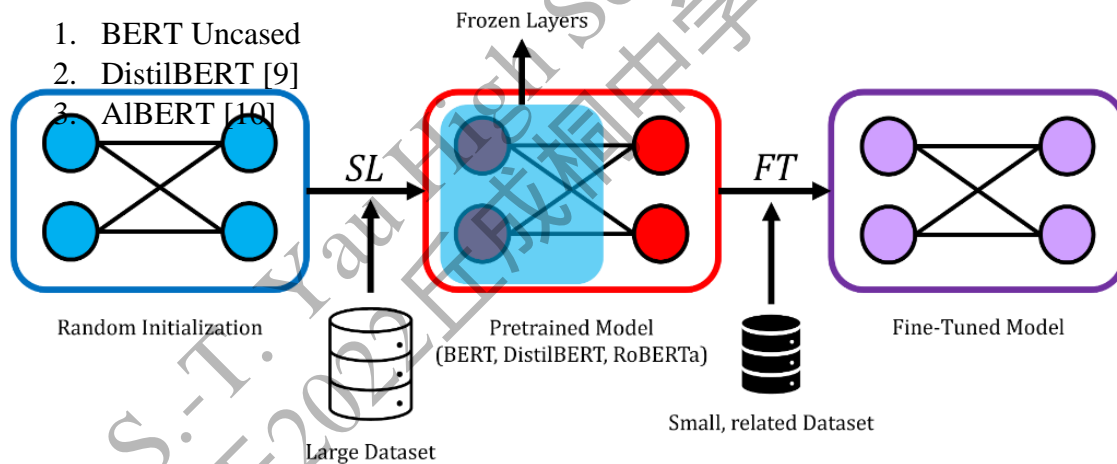


Figure 6: Diagram of how transfer-learning works

We see that there is a clear distinction between the different types of BERT models, however they all tend to peak at around 80%. BERT has the highest accuracy of 81%, however the other two models are not far behind, with 80% for DistilBERT and 79% for AIBERT. As distilBERT has a much smaller architecture than the rest, allowing it to be more efficient, and more easily trained, thus it was chosen for our implementation.

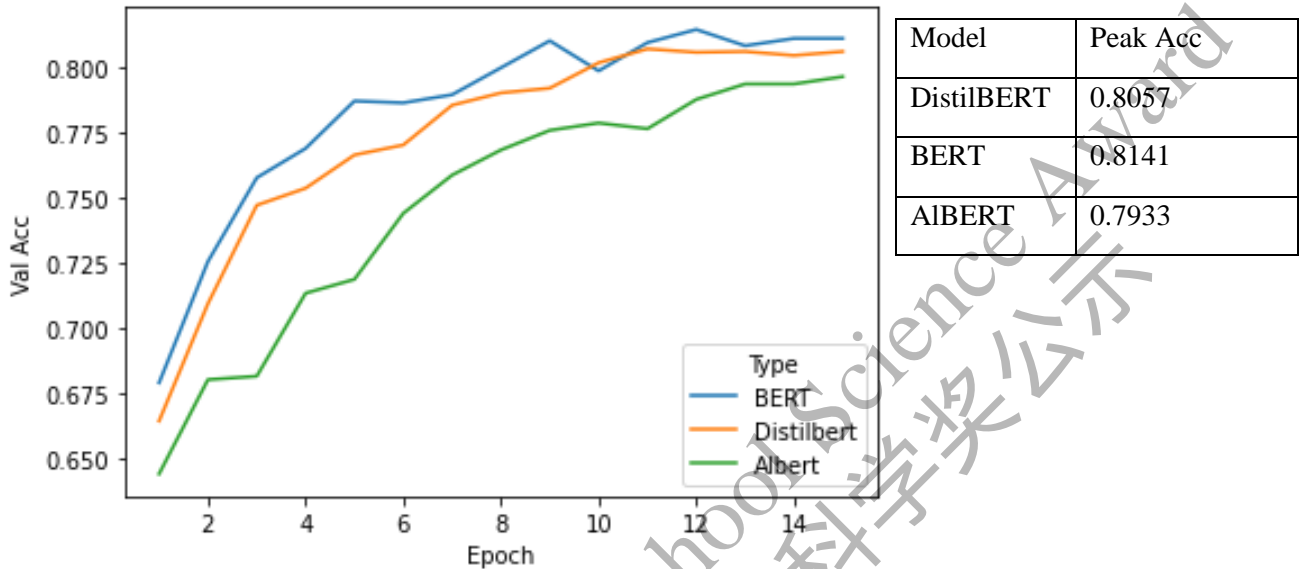


Figure 7: Graph of validation accuracy against epoch for the BERT models

Clustering

We clustered the background info using the KMeans algorithm [11] [12] [13]. It is an unsupervised clustering algorithm which is used to cluster our data into K number of clusters. It iteratively reduces the distances between each data point and its centroid to find the best solution for all data points. At the end of running KMeans, we will have K number of cluster centroids and all the data points will be classified into the clusters. KMeans is implemented as follows:

1. Choose K points from the dataset as the starting centroids, either at random or as the first K.
2. Calculate the Euclidean squared distance between each point in the dataset and the indicated K points (cluster centroids).
3. Using the distance determined in the previous step, assign each data point to the nearest centroid.
4. Take the average of the points in each cluster group to find the new centroid.
5. Repeat steps 2–4 for a set number of iterations, or until the centroids do not change.

Mathematically, the second step entails splitting the observations based on the Voronoi diagram produced by the centroids.

$$S_i = \{x_p: \|x_p - c_i\|^2 \leq \|x_p - c_j\|^2 \forall j, 1 \leq j \leq k\}$$

Let p and q are different document vectors, then the Euclidean squared distance is given by

$$d(p, q) = (p - q)^2$$

The finding of new centroids from each respective clusters formed is implemented using this equation:

$$c_i = \frac{1}{|S_i|} \sum_{x_i \in S_i} x_i$$

where S_i is the set of all points assigned to the i th cluster. In this equation, we are assigning the new centroid as the mean of all of the points in the cluster.

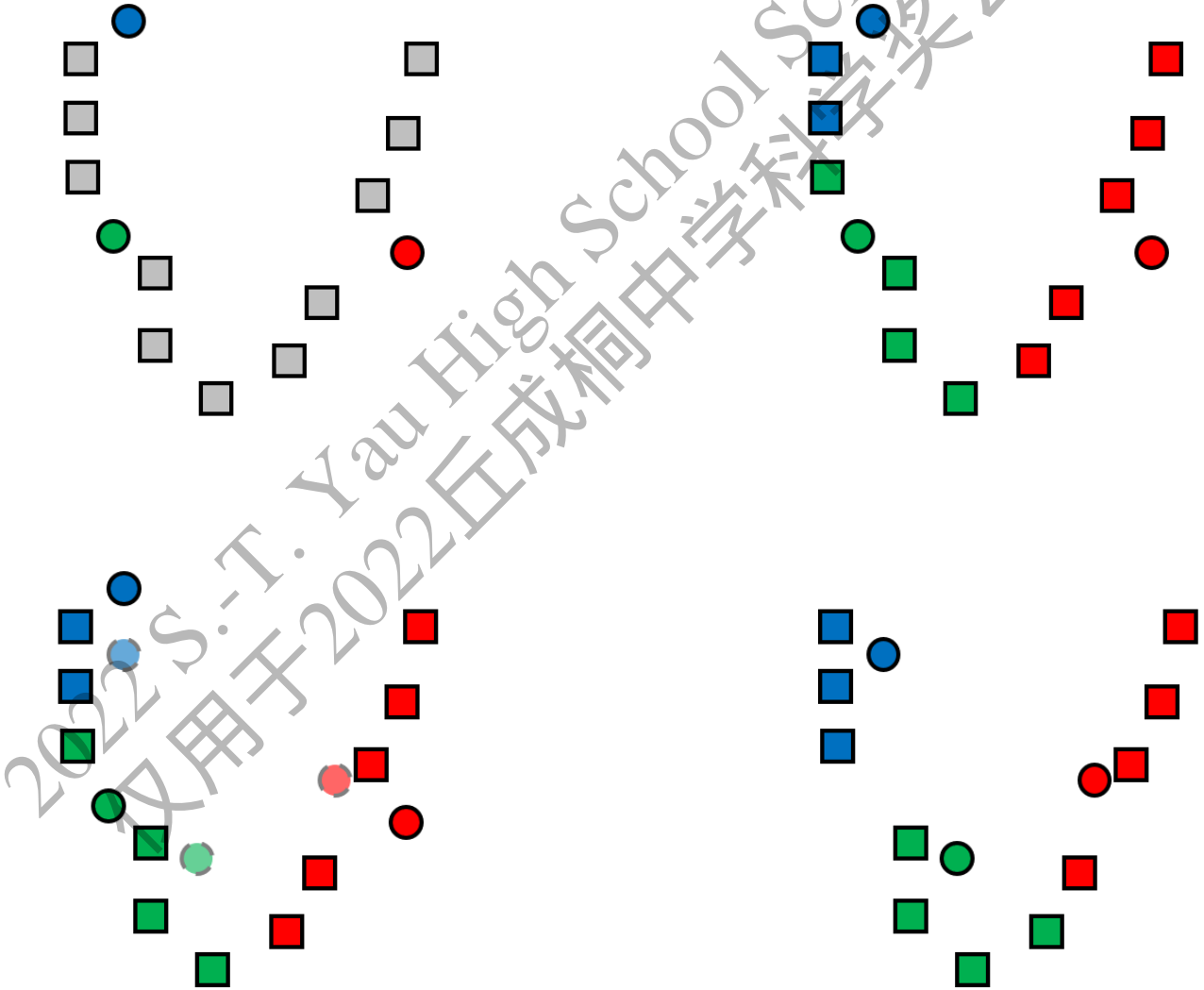


Figure 8: A visual representation of the steps for KMeans

The K-means algorithm minimizes an objective function, which in this case is the squared error function which is given by

$$J = \sum_{i=1}^k \sum_{j=1}^n (x_i - c_i)^2 = 1$$

Feature Extraction

Firstly, to benchmark the estimators, a dictionary vectorizer was used by doing feature extraction. Feature extraction is a technique that aims to transform raw data to derived values. The new features can summarize the large volume of text efficiently to link it to a specific law, thus making our model more efficient and faster. This could be accomplished using Bag-of-Words model, which is a tool for feature generation. For Bag-of-Words, text is represented as a list of separated words. From this initial step, analysis is done, such as the frequency of the words, to vectorise the documents. We decided to use TFIDF. In our implementation of feature extraction, terms that appeared in at least 50% of the documents, or not present in at least 5 documents are ignored. This allowed each point to have 20892 features, that could be used to cluster them. These features are used as the dimensions for a document vector.

To begin, TF-IDF can be divided into two parts: TF (term frequency) and IDF (inverse document frequency). Term frequency works by examining the frequency of a certain term in relation to the document. TF is given by

$$tf(t) = \frac{nf(t)}{nt}$$

where t is the term we are looking to measure the commonness of, nt is the number of terms in a document and $nf(t)$ is the number of times the term ' t ' occurred in the document.

Inverse document frequency examines how prevalent (or rare) a term is in the corpus. The IDF is computed as follows:

$$idf(t) = \log\left(\frac{1+n}{1+df(t)}\right) + 1$$

where t is the term we are looking to measure the commonness of and n is the number of documents in the corpus. $df(t)$ is just the number of documents that contain the term t . The ones are added to prevent division of 0.

By multiplying TF and IDF together as such,

$$tfidf(t) = tf(t) \cdot idf(t)$$

,we can get our final TF-IDF score. The higher the TF-IDF score the more important or relevant the term is; as a term gets less relevant, its TF-IDF score will approach 0.

Dimensionality reduction using LSA

To solve the problem of there being too many features, LSA [14], a technique of analysing relationships between documents, was used. It can be used to reduce the number of features, and thus the number of dimensions of the document vector. This allows KMeans to be more stable and faster. LSA uses a document-term matrix, which specifies the occurrences of words in documents. The cells of the matrix contain the weight of each term in a document. The matrix can be converted to three matrices, context-term, context-context and document-context, this is known as Singular Value Decomposition (SVD). In terms of dimensions, multiplying the three matrices returns us to the form of our original matrix, with the context dimension basically vanishing. Firstly, a “singular values” is an array of numbers which represent how much each of the topics explains a certain data. We will represent it as Σ , the original matrix as M , U as the document-context matrix and v^T as a transposed version of the term-context matrix. The context dimension was not in our original matrix.

$$M = \sum_i \sigma_i u_i \otimes v_i^T$$

where for the i th column in u and column in v^T which are vectors.

The summation is of the outer product of the two vectors, weighted by the singular value σ_i . The outer product of two vectors of the forms $(m,)$ and $(,n)$, yields a matrix with the shape (m,n) . In other words, the new matrix contains every conceivable product of any two values in the two vectors. The singular value not only weights but also arranges the total, because the values are sorted in decreasing order, with the first unique value always being the highest.

Finding most similar case

A function was implemented, which returns the closest points based on the text. This was achieved by first vectorizing the text with TF-IDF then performing an LSA reduction to it. We have considered 2 distance algorithms for obtaining closest cases to the centroid. Squared Euclidean distance. However, this does not pose a significant detriment to our system. Levenshtein distance was initially considered, however, our dataset was too large for use of Levenshtein distance. From that, the squared Euclidean distance is calculated from the given text and based on the input on the max squared Euclidean distance, the closest points are chosen.

Modified elbow method

To give the user the accurate and relevant acts, while not giving them too many acts that can confuse them, a modified elbow method is used. In the proposed model, a custom training is used that changes the size of the clusters, and then classify the user’s text to the individual clusters. The added layer of a classifier ensures that the accurate acts would be returned to the user, even with the added modifications and use of colloquial terms. The background information from the court proceedings is used in the clustering. Noise is then added and use new data to train the classification model.

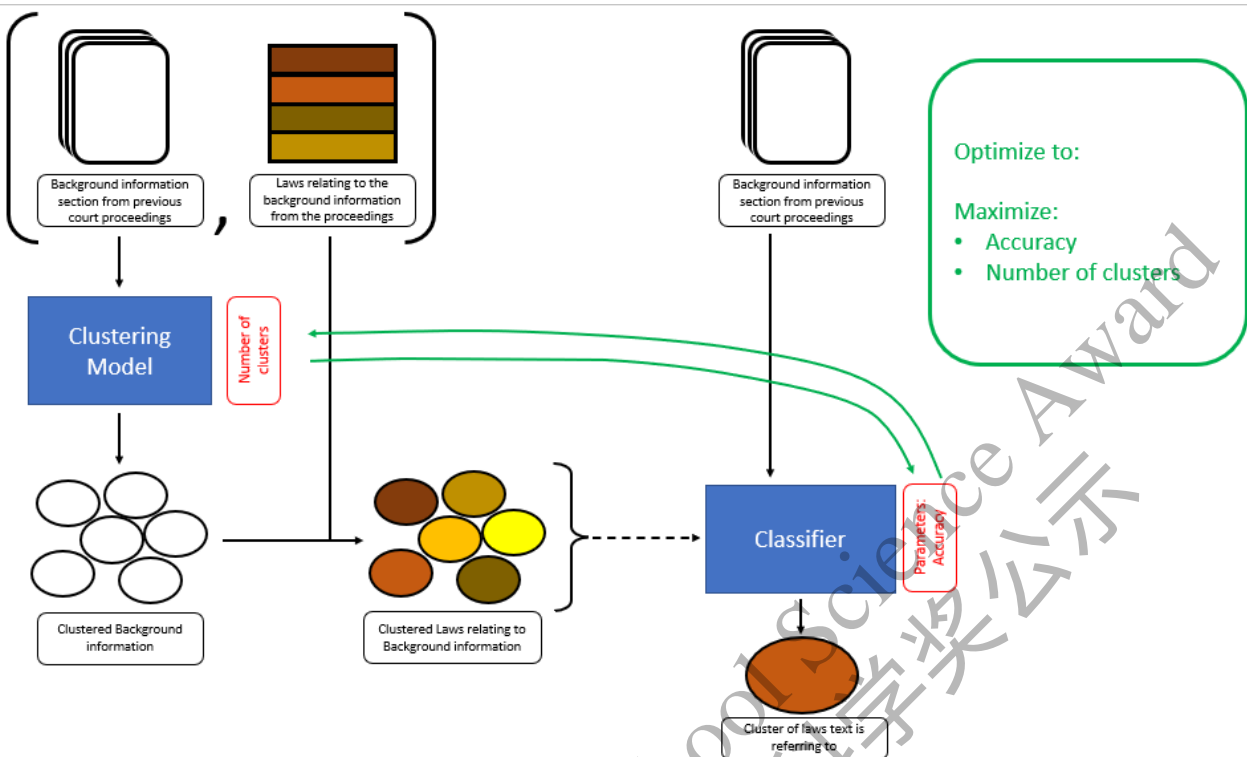


Figure 9: Proposed model to optimize number of laws to return to users

In our implementation, the Supreme Court Decisions are scraped to find the acts in them. Then, they are fed to the clustering model, multiple times, with a different number of clusters. Along with this, a function was implemented to get the nearest cases using the vectorization done for KMeans. This will thus allow us to know which laws are similar, or which laws are generally seen together. The clusters are used to train the classifier of multiple BERT models to classify it to a cluster. Separately, the data from the Community Justice Center were used to train the classifier of multiple BERT models to classify text to a region of law. When a user inputs some background info, the CJC trained classifier classifies it to a region of law. The SCD trained classifier classifies it to a cluster, allowing us to get what acts the case may have. Finally, using the implemented function, similar cases could be identified. This info is outputted to the user.

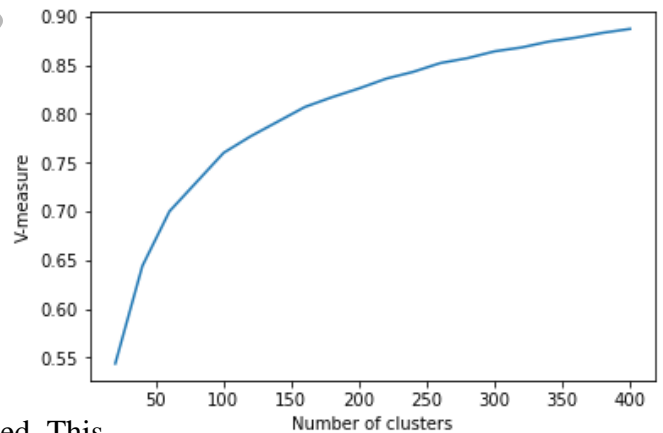


Figure 10: Graph of V-measure across clusters

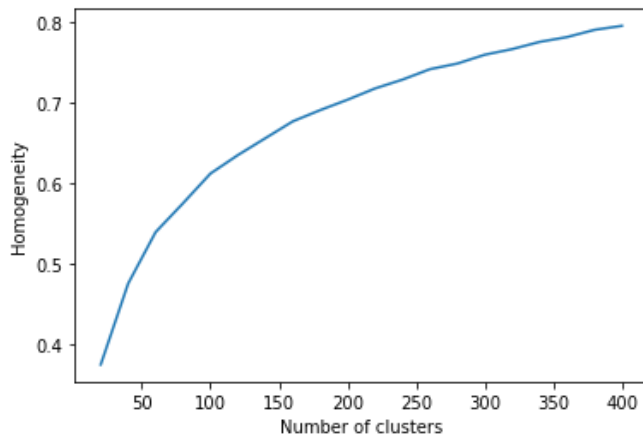


Figure 11: Graph of Homogeneity across clusters

Homogeneity [15] is the measure of how similar the data points are in a cluster. As seen in Figure 8, as the number of clusters increases, the homogeneity also increases. This makes sense, as since the number of clusters increases, there will be more centroids. This will result in only the closest points forming a cluster.

V-measure [16] is calculated using homogeneity and completeness, which is how many points of the same class are in the same cluster. This is a better measure than homogeneity, as it takes in account of the classes of points. As seen in Figure 9, as the number of clusters increases, the V-measure also increases. However, V-measure

increases as a decreasing rate, suggesting that a point exists, where there is high similarity in clusters, but also high accuracy in the classifier, which thrives with less clusters.

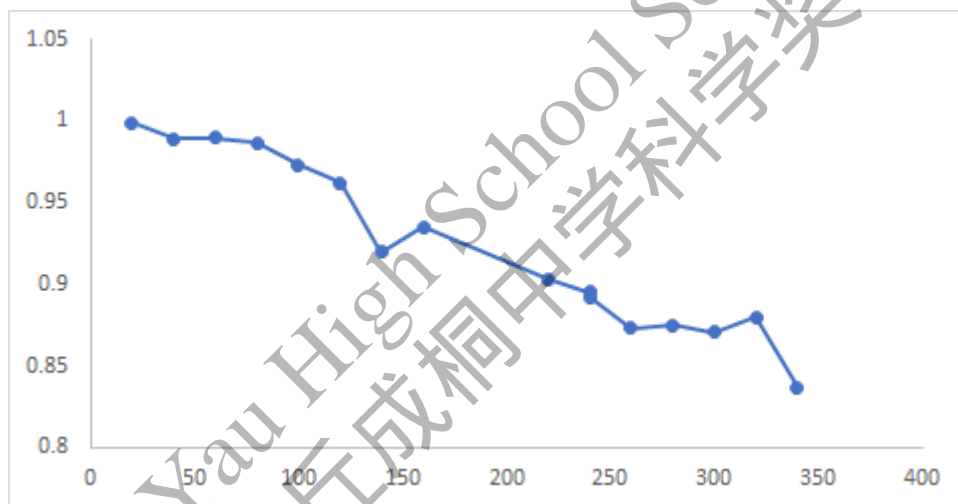


Figure 12: Graph of accuracy against number of clusters

As can be seen in the figure above, as the number of clusters increase, the accuracy drops linearly. This is because as the number of clusters increases, the similarity among these clusters also increases, as the clusters become more specific. This could lead to the classifier incorrectly putting the text in a similar, yet not exactly accurate cluster. This is in contrast with the clusters being general initially.

Telegram API

The telegram interface was implemented using telegram's Conversation Handler, taking command input, and returning the relevant documents requested by the user. The user will first enter in a command that is formatted as follows: “/find [background_info]”. The resulting data will be parsed from the input command and passed to our model. The output, relevant sections/acts/laws, will then be sent by our bot as a telegram message. The telegram interface is hosted on a Lenovo Flexpad Laptop, with Intel i5 core processor.

DISCUSSION

Our model is the first of its kind which is tailored for Singapore Law. The implementation proposed is also novel and we believe allow such a useful and accurate output.

We have hence chosen 160 clusters as the optimal number for our system. 160 performs with the optimal V-measure and Homogeneity scores, in addition to accuracy, as can be seen from Figure 10-12.

Our approach to applying Natural Language Processing in Law uses a novel system that is specifically adapted for processing legal texts. Our system is developed entirely from our own findings, so as to propose a method independent of other existing works. Our system is designed for user friendliness, as opposed to most other research. This means that our work is directly applicable to real life situations, and not limited to research.

RECONCILIATION

Background Information on Singapore court cases from 2016 to 2020 was obtained from Singapore Community Justice Centre, as well as through the scraping and parsing of public records. The data consists of summarizations of court cases categorized by 13 case types. A BERT language model was used in conjunction with Noisy student training to train with data from Singapore Justice Centre cases. The data was reorganized into labels for 10 case types, and data augmentation was applied for equalization of data. Our data was trained using several different BERT models to find the optimal choice. We have used Kmeans, an algorithm for obtaining a centroid and a list of documents similar to the centroid. Random points are taken as centroids of clusters, and documents are added to a cluster based on Euclidean distance. However, centroids can be selected such that they contain only the document they originate from. Hence, we have increased our n_init , number of runs with independent random initiations, to 20. We have implemented a modified algorithm system, where educated adjustments are made to the clustering algorithm based on previous results, optimizing our algorithms and obtaining the most accurate and specific laws for the user. The cluster data was trained with a BERT classifier for finding label n given a document.

Further improvements

An improvement is to expand our datasets, and work more closely to Singapore courts. This will allow us to increase the accuracy of our model as more data increases the probability that it contains useful information. It will also reduce the overfitting of our model, making it more robust.

Another extension to our research can be made by implementing versions of the bot for other countries, that may have different laws compared to Singapore law. This will allow us to reach a wider scope of lawyers and people uneducated in law. Expanding our project will also allow us to understand how to manage large system management. However, the modified elbow method will allow us to adapt our solution more easily.

Our current implementation only returns a list of Acts, rather than the specific section from the act. This is due to the fact that each Act is distinct from another, and thus, it is very simple to identify which Act is related based on the case. A section is a discrete component or provision of a legal code or collection of laws that frequently establishes a specific legal obligation. Sections are more specific

than acts, and can lead to less time loss. However, we are unable to do this due to the data not being distinct enough to identify each section accurately.

ACKNOWLEDGEMENTS

We would like to acknowledge and thank our mentor, professor Lim How Khang, for supporting and guiding us along the way. We would also like to thank the Community Justice Centre (CJC) for being able to provide us with their data, as well as our teacher mentor Mr. Lim Teck Choow.

REFERENCES

- [1] Devlin, Jacob, et al. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. arXiv, 24 May 2019. arXiv.org, <http://arxiv.org/abs/1810.04805>.
- [2] Telegram APIs. <https://core.telegram.org/>. Accessed 16 Aug. 2022.
- [3] NLTK :: Natural Language Toolkit. <https://www.nltk.org/>. Accessed 16 Aug. 2022.
- [4] Singaporean Legal Materials. <http://www.commonlii.org/sg/>. Accessed 16 Aug. 2022.
- [5] Home - Singapore Statutes Online. <https://sso.agc.gov.sg/5443/>. Accessed 16 Aug. 2022.
- [6] Vaswani, Ashish, et al. Attention Is All You Need. arXiv, 5 Dec. 2017. arXiv.org, <http://arxiv.org/abs/1706.03762>.
- [7] Srivastava, Nitish, et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." Journal of Machine Learning Research, vol. 15, no. 56, 2014, pp. 1929–58. jmlr.org, <http://jmlr.org/papers/v15/srivastava14a.html>.
- [8] Raffel, Colin, et al. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. arXiv, 28 July 2020. arXiv.org, <http://arxiv.org/abs/1910.10683>.
- [9] Sanh, Victor, et al. DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter. arXiv, 29 Feb. 2020. arXiv.org, <http://arxiv.org/abs/1910.01108>.
- [10] Lan, Zhenzhong, et al. ALBERT: A Lite BERT for Self-Supervised Learning of Language Representations. arXiv, 8 Feb. 2020. arXiv.org, <http://arxiv.org/abs/1909.11942>.
- [11] Ball, Geoffrey H., and David J. Hall. Isodata, a Novel Method of Data Analysis and Pattern Classification. Stanford Research Institute, 1965.
- [12] MacQueen, J. "Some Methods for Classification and Analysis of Multivariate Observations." Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics, vol. 5.1, Jan. 1967, pp. 281–98. projecteuclid.org, <https://projecteuclid.org/ebooks/berkeley-symposium-on-mathematical-statistics-and-probability/Proceedings-of-the-Fifth-Berkeley-Symposium-on-Mathematical-Statistics->

[and/chapter/Some-methods-for-classification-and-analysis-of-multivariate-observations/bsmsp/1200512992.](#)

[13] Scikit-Learn: Machine Learning in Python — Scikit-Learn 1.1.2 Documentation. <https://scikit-learn.org/stable/index.html>. Accessed 16 Aug. 2022.

[14] Deerwester, Scott, et al. “Indexing by Latent Semantic Analysis.” *Journal of the American Society for Information Science*, vol. 41, no. 6, 1990, pp. 391–407.

[15] Jain, Brijnesh J. Homogeneity of Cluster Ensembles. arXiv, 8 Feb. 2016. arXiv.org, <http://arxiv.org/abs/1602.02543>.

[16] Rosenberg, Andrew, and Julia Hirschberg. “V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure.” *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, Association for Computational Linguistics, 2007, pp. 410–20. ACLWeb, <https://aclanthology.org/D07-1043>.

[17] Figure 3. Alammar, Jay. *The Illustrated GPT-2 (Visualizing Transformer Language Models)*. <https://jalammar.github.io/illustrated-gpt2/>. Accessed 16 Aug. 2022.

[18] Bird, Steven, Edward Loper and Ewan Klein (2009), *Natural Language Processing with Python*. O’Reilly Media Inc.