Stance Detection on Forum Debates

Jack Cheng Ding Han

Supervisor: Andreas Vlachos
Module: COM3610 Dissertation Project

Department of Computer Science
University of Sheffield

May 2018

This report is submitted in partial fulfilment of the requirement for the degree of Master of Computing by Jack Cheng Ding Han.

All sentences or passages quoted in this report from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations which are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this project and the degree examination as a whole.

Name: Jack Cheng Ding Han

Signature: 

Date: Wednesday 9th May, 2018

Abstract

The project demonstrates supervised learning approaches for the classification task of stance detection where the data-sets are sourced from posts on online debate forums. Initially, two different kinds of approaches had their performances compared:

- Support Vector Machines (SVMs)
- Long Short-Term Memory (LSTM)

Findings from the experiments with these approaches show the proportion of classes affecting the performance of a given classifier. Based on the work of Augenstein et al. [1], the detection of the stances of forum posts made about topics that the classifiers have not seen before – an unseen target – was also explored. Building upon this, the possibility of correlation between the textual similarity of two topics' debates and classifier performance when one of the two is exclusively used as testing data was investigated as well.

Acknowledgements

I'd like to thank the following people: My parents who put me on a path towards a STEM career, the IT teacher who introduced me to robotics and programming in year 8, my year 10/11 mathematics teacher S. Wake, my sixth-form teachers C. Bower, T. Westby and Alan. Dr. M. Stevenson, Prof. R. Gaizauskas, Prof. P. Green, Prof. J. Barker, and Dr. A. Vlachos who all led me to pursue the field of machine learning.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Online debate forums are web applications that enable users to make threads to debate topics and posts responses to said threads. With so many users taking part, these systems can yield very large corpora of text. A typical online debate forum, `CreateDebate`*, alone contains at least fifty-seven thousand different debates† across fifteen distinct topics as of 2018. Thriving with discourse, disagreement, and opinion, online debate forums provide many opportunities to perform natural language processing tasks. One example of these tasks is stance detection.

According to Krejzl et al. [2], stance detection is detecting whether a given body of text written by an author is in favour of a given target, against the given target or neither. Stance detection is an example of a classification task where a classifier must take a body of text and put it into a class corresponding to a stance.

This project will have the bodies of text be the forum posts. The targets of each body of text will be the topics of the debate thread their respective post belongs to. For example, in a debate thread where the topic is abortion, the target of the body in each post will be abortion. Although some of the previous work mentioned in chapter 2 such as the investigations by Sridhar et al. [3], have other users and other posts be the targets, extending the idea of stance to model disagreement.

Stance detection is closely to sentiment analysis. It differs in that the target need not always be mentioned in the text and an author can be in favour of a target but not express or even intend positive sentiment whilst doing so. Figure 1.1 shows two forum posts about the topic of marijuana legalisation exemplifying these differences. Neither mentions marijuana legalisation. The first forum post is in favour of marijuana legalisation but the statement is objective with implicit sentiment. The second forum post is also in favour of marijuana legalisation and unlike the first post. Although the second post gives sentiment, it is negative.

*`http://www.createdebate.com/`

†2378 search pages, 24 debates per page

> "Harry J. Anslinger of the Federal Bureau of Narcotics conspired with William Randolph Hearst, a stakeholder in the timber industry, and the Du Pont chemical company, the inventors of nylon, to destroy the hemp industry so it would not threaten their sources of wealth."
>
> "We are not harming anyone by doing this, the government is being very tyrannical by banning cannabis. They say cannabis is a gateway drug. Well I say banning it is the gateway for our country to become a dictatorship. If these merciless and heinous abuses continue, I won't stay under the rule of these cruel, evil and heartless oligarchs."

Figure 1.1: Stance is not necessarily the same as sentiment.

The classifiers that have been implemented for this project all utilise supervised learning techniques. Classifying a forum post aimed a particular target would usually require a classifier to be trained on other posts aimed at the target which have already been pre-labelled with stances, said target is known as a *seen target*.

On the World Wide Web, users have the capability spread new ideas, find new topics to discuss and debate, or give new responses. Constant changes such as these give rise to serious complications such as the nonexistence or lack of access to pre-labelled posts about a given topic. A topic for which none of the posts that have been made about it are already pre-labelled with stance or where none of the pre-labelled posts are readily available to the classifiers would be an *unseen target*.

A workaround that can be put in to consideration is for the classifiers to be trained on forum posts aimed at targets that are related but may not be identical to the unseen target. Suppose the classifiers are to classify posts aimed at new political candidate; debates about the candidate may be too recent to have been pre-labelled but issues that concern the candidate could be decades old enough for them to possess pre-labelled posts. Since the classifiers cannot be trained on pre-labelled posts about the candidate, they could instead be trained on pre-labelled posts in debate threads where the topics are the issues that concern the candidate.

The project consists of two experiments, implemented in Python, that involve comparing stance detection approaches utilising a Support Vector Machine (SVM) to those that utilise Long Short-Term Memory (LSTM). The experiment described in section 5.1, is intended to find the best performing classifiers using aforementioned approaches. The second experiment described in section 5.2 is intended to see how well classifiers are able to perform when tested on posts that are about topics which are unseen targets. From the second experiment, there can be further investigations into classifier performance and how it is affected by the relatedness between a pair of topics (used as the seen and unseen targets of the classifier).

1.1 Report Overview

1. Chapter 2 focuses on existing ideas and research.
    (a) Section 2.1 reports previous natural language processing studies that have used posts from online forum debates as data-sets.
    (b) Section 2.2 has in-depth explanations of the existing ideas and tools that were vital to the project:
        i. Section 2.2.1 explains the intended baseline approaches.
            A. Section 2.2.1 explains the word vector model, how it works and why it was chosen.
            B. Section 2.2.1 explains the classification method, why it was chosen, previous work using the method and how the method works.
        ii. Section 2.2.2 explains the more recent approaches utilising Long Short-Term Memory.
    (c) Section 2.3 explains how the relatedness of topics will be calculated.
2. Chapter 3 explains rigorous metrics to judge the performance of the classifiers.
3. Chapter 4 explains the structure of the files in the raw data-set.
4. Chapter 5 explains the two experiment setups within this project.
5. Chapter 6 shows prominent and notable results in the experiments conducted
6. Chapter 7 are the conclusions made by drawing upon the findings.

Chapter 2

Literature Review

2.1 Natural Language Processing on Forum Debates

The use of forum debates as a data-set for an NLP task was previously done by Shi et al. in 2013 [4] using Chinese-language posts in the two-fold task of detecting the topics of threads as well as the sentiment of individual posts within each thread.

To detect topics, a K-means clustering algorithm was used, which is a form of unsupervised machine learning. The topic detector had to assign debates with a label, debates with the same label were regarded as having the same topic. The challenges that had to be looked at for this task included the difficulty in clustering very short bodies of text which was usually the case for the opening posts for the debates in their data-set, to overcome this, the text of the opening posts were combined with all of the replies on the first page and treat this combination as a document [4, p.149]. Afterwards, each document had to be converted into a vector where each of the components corresponded to a term and the value of each component was Term Frequency-Inverse Document Frequency (TF-IDF).

According to D. Jurafsky and J.H. Martin, TF-IDF is the product of term frequency – which is how often a term appears in an individual document, and inverse document frequency – which is a value that is larger for a terms that appear in fewer documents. [5, Chapter 15]. After clustering took place (using cosine similarity as a measure of closeness) each cluster had to be assigned a topic description, and in order to achieve this, a technique similar to the aforementioned TF-IDF was used. The best terms to describe a cluster, all of which were nouns, were those that appeared the most often in an individual cluster cf. term frequency but less often in other clusters, cf. inverse document frequency [4, p.149]. To detect sentiment, they had to construct what they referred to as a Probability Word-List, with the purpose of retrieving words by which the sentiment opinions are differ-

entiated from one another. They classified based on a value for each thread that is computed using distance between sentiment words and the nouns from the aforementioned topic description in conjunction with the semantic features from their Probability Word-List. This was a ternary classification task; posts had to be put into the classes of positive, neutral or negative. This proposed approach using a Probability Word-List managed to outperform Support Vector Machines [4].
The data-set that is used in this current project, `CreateDebate`, was previously among those that were used by Sridhar et al. in 2015 [3] in order to investigate the modelling of disagreement in online debate forums. The ramifications of choosing whether to model collectively or locally as well as the choice between post-level modelling and author modelling were all explored. The modelling of disagreement is difficult because it relies on the assumption that stance alternates between replies, which is not always the case. The approach that Sridhar et al. proposed to overcome such limitations included the prediction of *reply link polarity* along with stance. Sridhar et al. described two variants of reply link polarity:

1. Textual disagreement is where replying posts are encoded as having agreement or disagreement with the text of the post it is replying to.

2. Stance disagreement, which is stance where target is the topic of the thread. Users could have the same stance for a topic but disagree with each other.

2.2 Stance Detection Approaches

2.2.1 Non-Conditional Approaches

Vectorisation

The techniques for classification described in later on in section 2.2.1 require the inputs to be vectors with numerical values. Data in the form of text such as forum posts must go through preprocessing stages to convert it into suitable input. Conversion of words or sequences of words into a vector requires a word vector model (WVM). The word vector model that is to be used with the SVM-based approaches is a continuous Skip-Gram (SG) model. The Skip-Gram word vector model was developed by Mikolov et al. [6]. Within the Skip-Gram model, n-tuples called c-skip-n-grams* are a generated. These are a generalisation of n-grams to allow tokens to be skipped with a parameter, c, which is known as a context window. For a given phrase, the set of c-skip-n-grams is the union of the sets of n-tuples where each token in the phrase is the first element and each second

*Guthrie et al. uses k to represent the context window, [7]. Mikolov et al. uses c [6].

element (context tokens) consist of tokens that are up to c places away (whether to the left or right) of the token. Guthrie et al. in 2006 gives the formula in fig. 2.1 to generate sets of c-skip-n-grams for each token w_i in the sequence $[w_1...w_m]$ [7].

$$\left\{ w_{i1}, w_{i2}, ...w_{im} \,\middle|\, \sum_{j=1}^{n} i_j - i_{j-1} < c \right\}$$

Figure 2.1: Generating a set of c-skip-n-grams.

For example, the phrase:

"How can you say such a thing?"

is tokenised into `['how', 'can', 'you', 'say', 'such', 'a', 'thing']`. From this, 2-skip-2-grams can be generated.

1. `'how'`
   (a) `('how', 'can')`
   (b) `('how', 'you')`
2. `'can'`
   (a) `('can', 'how')`
   (b) `('can', 'you')`
   (c) `('can', 'say')`
3. `'you'`
   (a) `('you', 'how')`
   (b) `('you', 'can')`
   (c) `('you', 'say')`
   (d) `('you', 'such')`
4. `'say'`
   (a) `('say', 'can')`
   (b) `('say', 'you')`
   (c) `('say', 'such')`
   (d) `('say', 'a')`

5. `'such'`
    (a) `('such', 'you')`
    (b) `('such', 'say')`
    (c) `('such', 'a')`
    (d) `('such', 'thing')`
6. `'a'`
    (a) `('a', 'say')`
    (b) `('a', 'such')`
    (c) `('a', 'thing')`
7. `'thing'`
    (a) `('thing', 'such')`
    (b) `('thing', 'a')`

The set of these 2-skip-2-grams is the superset of 2-grams for the phrase but in addition to considering tokens immediately adjacent to each-other to create the 2-tuples, tokens that are two places away are away considered. Also note that tokens nearer to the beginning of the phrase such as `'how'` and `'thing'` get fewer 2-tuples generated for them because for the former, there are fewer tokens to the left and for the latter, there are fewer tokens to the right. In contrast, tokens nearer to the centre of the phrase have the most 2-tuples generated for them as they have at least 2 tokens that are one to two places away to both their left and right.

Vectorisation of the phrase is not yet complete as the tokens have not been converted into vectors with numerical entries. More steps follow the generation of the Skip-Grams. Using a provided vocabulary of words. Each token is converted to a simple One-Hot Vector with its length equal to V which is the size of the vocabulary. A One-Hot Vector is a vector where one and only one of the elements is equal to one and the rest of the elements are equal to zero. A neural network is created with:

- A $1 \times V$ input layer.
- A $1 \times d$ projection layer.
- A $C \times V$ output layer

For each token, the network is trained on its One-Hot Vector where the training labels for the each of the C columns in the output layer are the One-Hot Vectors of the input's context tokens. The network outputs the probabilities of the context tokens appearing near the input token. Mikolov et al. provides a loss function in fig. 2.2 with the objective of maximising the average log probability [6, p.2]. After training, the weights of the connections between the layers are adjusted,

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log \mathrm{P}\left(w_{t+j}|w_t\right)$$

Figure 2.2: Loss function for the Skip-Gram model.

the probabilities that have been outputted can be ignored and focus is put on the weights between the input layer and the projection layer represented by the matrix **W** which has V rows and d columns. Row y where $y < V$ can be extracted as a d-dimensional dense vector and this is the vector representation of the input token. The intuition behind the Skip-Gram word vector model is that words appearing in similar contexts (being surrounded by similar words), will produce vectors with close values to each-other because the probabilities given by the output layer will be similar. In this project, the Skip-Gram word vector model is implemented using the gensim* toolkit.

Classification

The intended baseline approach towards stance detection in this project is the use of Support Vector Machines which were invented by V.N. Vapnik [8]. Support Vector Machines are effective in text processing tasks e.g. it was shown by Pang et al. in 2002 that Support Vector Machines were able to surpass the performance of Naive Bayes classifiers for the task of sentiment analysis [9].

Küçük et al. in 2018 used Support Vector Machines to perform stance detection where the bodies of text were sourced from tweets written in Turkish and the targets were sports-related. A notable investigation done for the paper was what to include in feature set; unigrams, bigrams, hashtags, external links, emoticons and named entities. Emoticons for instance, area a non-alphanumeric means of communicating sentiment so stripping them away during preprocessing could lose valuable information. Nevertheless, their findings indicated that a joint feature set of unigrams, hashtags, and named entities was the most plausible approach [10].

*`https://radimrehurek.com/gensim/models/word2vec.html`

The most basic of Support Vector Machines are linear classifiers which find the optimal hyper-planes to separate the training data. This is achievable by searching for the direction that gives the maximum possible margin. A small subset of the training data, the titular support vectors, are used in finding the margin. These support vectors are members of the training data that would change the position of the optimal hyper-plane if they were to be removed because they lie exactly on the margins of the optimal hyper-plane.

Assume a binary classification task with two classes representing the stances "AGAINST" and "FAVOR". A hyper-plane $\mathrm{h}(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = 0$ must be designed that classifies the training data correctly [11, Chapter 3]. The distance of a given point to a hyper-plane is $z = \dfrac{\mathrm{h}(\mathbf{x})}{\|\mathbf{w}\|}$. Let H_1 be the hyper-plane for the "FAVOR" stance such that $\mathrm{h}(\mathbf{x}) = +1$ and H_2 be the hyper-plane for the "AGAINST" stance such that $\mathrm{h}(\mathbf{x}) = -1$

Hyper-plane H_0 is the situated between the two satisfying $\mathrm{h}(\mathbf{x}) = 0$. Since the distance of a given point to a hyper-plane is $z = \dfrac{\mathrm{h}(\mathbf{x})}{\|\mathbf{w}\|}$, the total distance between H_1 and H_2 would be $\dfrac{2}{\|\mathbf{w}\|}$ [11, Chapter 3].

In order to maximise the margin, $\|\mathbf{w}\|$ must be made as small as possible so long as H_1 and H_2 have no data-points in between them. This is a quadratic programming problem that can be solved using Lagrangian multipliers. The function $f = \dfrac{1}{2}\|\mathbf{w}\|^2$ must be minimised subject to $\mathrm{g}(\mathbf{x}) = y_i(\mathbf{w}\cdot\mathbf{x} - w_0) - 1 = 0$

A reformulation with a Lagrangian $\mathcal{L}$ and slack variable a has $\mathcal{L}(\mathbf{x}, \alpha) = f(x) - ag(\mathbf{x})$ and $\nabla(\mathbf{x}, a)$

The first function can be generalised as $\mathcal{L}(\mathbf{x}, \alpha) = f(x) + \sum_i a_i g_i(\mathbf{x})$

Substituting the original expressions for f and g in gives the primal problem where $\mathcal{L}_P = \dfrac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{l} a_i y_i(\mathbf{w}\cdot\mathbf{x} + w_0) + \sum_{i=1}^{l} a_i$ must be minimised subject to all a_i greater than or equal to zero.

The dual problem is maximising $\mathcal{L}_D(a_i) = \sum_{i=1}^{l} a_i - \dfrac{1}{2}\sum_{i=1}^{l} a_i a_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j)$ subject to $a \geq 0 \wedge \sum_{i=1}^{l} a_i = 0$. Because data cannot always be easily separated by straight lines or their higher dimensional analogues, Support Vector Machines allow for non-linear classification to take place by using a similarity function that is referred to as a kernel. The kernel transforms the data-set using a mapping so that data-points within different classes become linearly separable. Figure 2.3 shows that the inclusion of a kernel function K to deal with non-linear classification can be elegantly represented when Support Vector Machines are formulated as a quadratic programming problem.

The implementation of Support Vector Machines that are used by this project

$$\mathcal{L}_D = \sum a_i - \frac{1}{2} \sum a_i a_j y_i y_j K\left(\mathbf{x}_i \cdot \mathbf{x}_j\right)$$

Figure 2.3: The SVM function to be optimised with kernel K.

are from the scikit-learn* machine learning library.

2.2.2 Conditional Approaches

The Non-Conditional approaches that have been implemented are compared to the recently proposed approaches described by Augenstein et al. [1] which utilise Long Short-Term Memory (LSTM) networks to perform stance detection on Tweets — these approaches can be adapted to other bodies of text such as forum posts. Long Short-Term Memory is a neural network architecture developed by Hochreiter and Schmidhuber [12]. A network of these LSTM blocks can encode text (via word embedding techniques that map text to vectors of real numbers). Two such approaches using LSTM were `TweetOnly` (or `PostOnly`, in this project) and `Concat`. Much like the SVM approach, the sole LSTM of the former approach only extracts features from the bodies of text. In contrast, `Concat` uses two LSTMs; one to encode the body of text and another to encode the target *independently* as vectors of the same dimension [1, p.877-880]. LSTM networks are formed of LSTM blocks which have four layers (three sigmoid, one hyperbolic tangent):

Input vector at time t: $\mathbf{x}_t$

Output vector at previous time $t-1$: $\mathbf{h}_{t-1}$

Matrix of the above: $\mathbf{H} = \begin{bmatrix} \mathbf{x}_t \\ \mathbf{h}_{t-1} \end{bmatrix}$

Entry-wise product: $\odot$

Sigmoid function: σ

Hyperbolic tangent: tanh

Weight matrices: $\mathbf{W}^i$, $\mathbf{W}^f$, $\mathbf{W}^o$ and $\mathbf{W}^c$

Bias vectors: $\mathbf{b}^i$, $\mathbf{b}^f$, $\mathbf{b}^o$ and $\mathbf{b}^c$

Forget gate: $\mathbf{f}_t = \sigma\left(\mathbf{W}^f \mathbf{H} + \mathbf{b}^f\right)$

*`http://scikit-learn.org/stable/modules/svm.html`

Input gate: $\mathbf{i}_t = \sigma\left(\mathbf{W}^i\mathbf{H} + \mathbf{b}^i\right)$

Output gate: $\mathbf{o}_t = \sigma\left(\mathbf{W}^i\mathbf{H} + \mathbf{b}^i\right)$

Cell state: $\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh\left(\mathbf{W}^c\mathbf{H} + \mathbf{b}^c\right)$

Output vector: $\mathbf{h}_t = \mathbf{o}_t \odot \tanh\left(\mathbf{c}_t\right)$

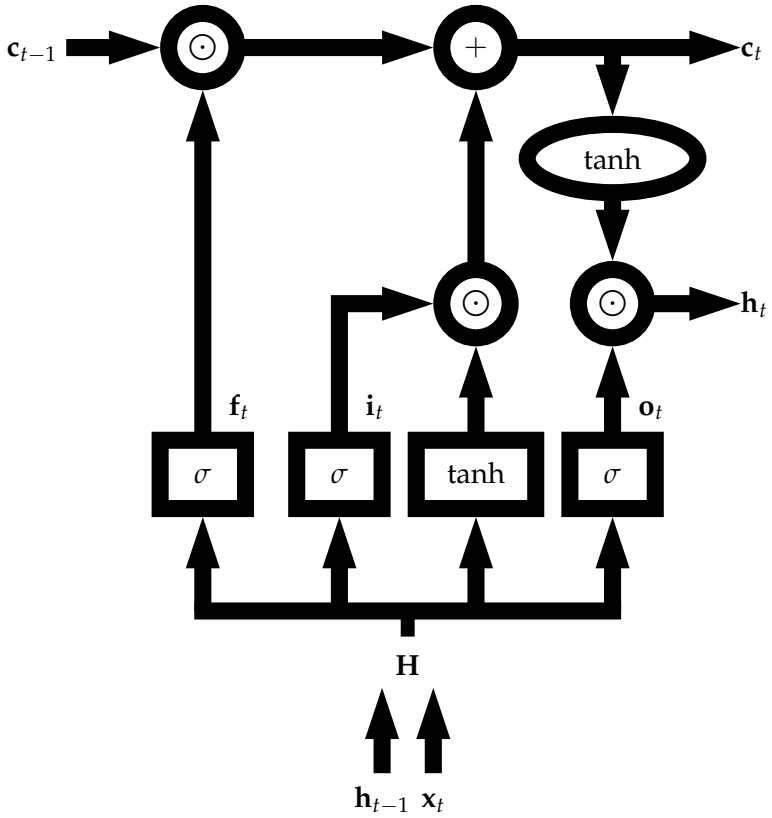


Figure 2.4: LSTM block.

All of the baseline approaches were compared to three approaches described as "Conditional Encoding", which was developed by Rocktäschel et al. [13] to determine how two pieces of text were related semantically. Two LSTMs are used; the first LSTM encodes the first sentence and the second LSTM encodes the second sentence. The task was to use conditional encoding determine if the two sentences contradicted each other, were not related at all or that the first sentence (the premise) entailed the second sentence (the hypothesis)[13, p.1]. What sets this network of two LSTMs apart from the network utilised in the baseline `Concat` approach, is that the two LSTMs are *not independent*; the initial cell state of the second LSTM is in fact initialised by the last cell state of the first LSTM [13, p.3].

As demonstrated by Augenstein et al. [1, p.878], Conditional Encoding can be adapted for the task of stance detection. Two of these conditional approaches were `TarCondTweet` where the first LSTM encodes the body of text and the second encodes the Target and `TweetCondTar` where the first LSTM encodes the Target and the second LSTM encodes the body of text.

The third conditional approach is `BiCond` (Bidirectional Encoding Model). Bidirectional LSTMs were developed by Graves and Schmidhuber [14] for the task of framewise phoneme classification. In the `BiCond` approach, the target and body of text are represented using two vectors for each of them; one from reading the target first and then the body of text (as with `TweetCondTar`) and one by reading the tweet first and then the body of text (just like in `TarCondTweet`).

The implementation of Long Short-Term Memory in this project comes from the Keras* neural network library using TensorFlow† machine learning framework as a backend.

2.3 Text Similarity

In order to investigate if the relatedness between the topic of the posts in the training data and the topic of the posts in the testing data could affect the performances of the classifiers, the debates available for each topic in the `CreateDebate`data-set have the bodies of the posts within them concatenated and then each of the these concatenations are themselves concatenated to form a large document. The similarity between the each topic's document would then computed. The algorithm chosen to do compute document similarity is Latent Semantic Analysis (LSA), developed by Deerwester et al. in 1990 for document retrieval [15].

After preprocessing such as tokenisation and stop-word removal is performed on each document. The list of tokenised documents is used to build a large term-by-document matrix $\mathbf{X}$ which has dimensions $|V| \times d$, where V is the vocabulary and d is the number of documents. Latent Semantic Analysis uses a technique called singular-value decomposition (SVD) for dimensionality reduction. SVD not just a way of lessening the work load; it is a method of finding the most important dimensions of a data set. The most important being the dimensions along which the data varies the most. Applying SVD on the term-by-document matrix factorises it into a product of three matrices [5, Chapter 16].

The three matrices that are factors of the term-by-document matrix $\mathbf{X}$ are:

- $\mathbf{U}$, an orthogonal $|V| \times d$. The columns in this matrix are called left singular

*`https://keras.io/layers/recurrent/`
†`https://www.tensorflow.org/`

vectors.

- **S**, a diagonal $d \times d$ matrix. The columns in this matrix are the titular singular values and are arranged from left to right in descending order.
- $\mathbf{V^T}$, the transpose of orthogonal $d \times d$ matrix **V**. The columns of matrix **V** are called right singular vectors.

Figure 2.5 shows term-by-document matrix **X** and its factorised form. The sum of

$$\underset{|V|\times d}{\begin{bmatrix} \\ \mathbf{X} \\ \\ \end{bmatrix}} = \underset{|V|\times d}{\begin{bmatrix} \\ \mathbf{U} \\ \\ \end{bmatrix}} \underset{d\times d}{\begin{bmatrix} \sigma_1 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & 0 & \cdots & 0 \\ 0 & 0 & \sigma_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_d \end{bmatrix}} \underset{d\times d}{\begin{bmatrix} \\ \mathbf{V} \\ \\ \end{bmatrix}}^T$$

Figure 2.5: Applying SVD to term-by-document matrix **X**

the squares of the singular values in matrix **S** should be equal to the total variance in matrix **X**. How much of each singular vector column in **U** and **V** accounts for the total variance is expressed by the corresponding singular value in matrix **S**.

The next step in singular-value decomposition is to only keep the top k singular values (typically, $k = 300$) and discard the rest. The resulting $|V| \times d$ dimensional matrix $\mathbf{X}'$ in fig. 2.6 is a least-squares approximation to the term-by-document matrix [5, Chapter 16].

$$\underset{|V|\times d}{\begin{bmatrix} \\ \mathbf{X}' \\ \\ \end{bmatrix}} = \underset{|V|\times k}{\begin{bmatrix} \\ \mathbf{U}' \\ \\ \end{bmatrix}} \underset{k\times k}{\begin{bmatrix} \sigma_1 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & 0 & \cdots & 0 \\ 0 & 0 & \sigma_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_d \end{bmatrix}} \underset{k\times d}{\begin{bmatrix} \\ \mathbf{V}' \\ \\ \end{bmatrix}}^T$$

Figure 2.6: Keeping only the top k singular values.

The implementation of Latent Semantic Analysis used by this project comes from the gensim* toolkit.

*`https://radimrehurek.com/gensim/models/lsimodel.html`

Chapter 3

Requirements & Analysis

3.1 Evaluating a Classifier

As mentioned in The Introduction, stance detection is a classification problem. The labels are stances and can be represented as the set $\{\text{AGAINST}, \text{FAVOR}\}$.

A classifier's performance is evaluated based on the following metrics:

- Accuracy
- Precision (one for each stance)
- Recall (one for each stance)
- F-measure (one for each stance)

Let $\text{predict}\,(\text{post})$ be the function a classifier uses to predict a post's stance and $\text{label}\,(\text{post})$ be the function for getting the actual stance of the post belonging to the testing set Test. Both of these aforementioned functions have the mapping $\text{Test} \mapsto \{\text{AGAINST}, \text{FAVOR}\}$.

The accuracy of each classifier is the ratio between the number of posts in the testing data that were labelled correctly to the size of the testing data.

$$\text{accuracy} = \frac{|\{\text{predict}\,(\text{post}) = \text{label}\,(\text{post})\,, \text{post} \in \text{Test}\}|}{|\text{Test}|}$$

Figure 3.1: The accuracy of a classifier.

The proportion of the most frequent stance in the training set is the baseline accuracy [16]. The rationale behind this is that most of the examples the classifier would learn from would be members of that stance.

The precision of a classifier given a stance is defined as the number of posts correctly classified as that stance divided by the number of posts that were classified as that stance.

$$\text{precision}_{\text{stance}} = \frac{|\{\text{predict}(\text{post}) = \text{label}(\text{post}) \wedge \text{label}(\text{post}) = \text{stance}, \text{post} \in \text{Test}\}|}{|\{\text{predict}(\text{post}) = \text{stance}, \text{post} \in \text{Test}\}|}$$

Figure 3.2: The precision of a classifier, given a stance.

The recall of a classifier given a stance is defined as the number of posts correctly classified as that stance divided by the number of posts in the testing data that actually belonged to that stance.

$$\text{recall}_{\text{stance}} = \frac{|\{\text{predict}(\text{post}) = \text{label}(\text{post}) \wedge \text{label}(\text{post}) = \text{stance}, \text{post} \in \text{Test}\}|}{|\{\text{label}(\text{post}) = \text{stance}, \text{post} \in \text{Test}\}|}$$

Figure 3.3: The recall of a classifier, given a stance.

The F-measure of a classifier given a stance is defined as the harmonic mean of the precision and recall.

$$\text{F-measure}_{\text{stance}} = 2 \cdot \frac{\text{precision}_{\text{stance}} \cdot \text{recall}_{\text{stance}}}{\text{precision}_{\text{stance}} + \text{recall}_{\text{stance}}}$$

Figure 3.4: The F-measure of a classifier, given a stance.

Chapter 4

Design

In the `CreateDebate`data-set, there are four directories for each of the four topics (abortion, gay rights, marijuana and Barack Obama) and in each of these directories files are two files for each post; `.data` files containing the post body and `.meta` files containing the author, and the stance label which can take one of two values; "-1" or "+1". The `.data` and `.meta` files of a post have names which are the same except for the extension, the names contain an alphabetical prefix identifying the debate and a numerical suffix identifying the post within said debate. As the data-set only provides two labels, classifying each post can be simplified to a binary classification task. As seen in fig. 4.2, in addition to stance, there were additional attributes that were made use of in the disagreement modelling investigations by Sridhar et al. [3]. These include the `PID` value to identify a post it is replying to and a `rebuttal` value.

```
It's not "life unworthy of life", it's actually "POTENTIAL life!"
Until the fetus can actually survive outside of the womb, it's a
living organ like any other organ inside the woman's body.  Once it
reaches the stage of about 20 weeks when it should stand a slight
chance of survival and maturing into a person, then it should be made
illegal.  But before that time, it only has the POTENTIAL, nothing
more.  A potential is just that.
```

Figure 4.1: `abortion/A4.data`, a file storing a body of text

```
ID=4
PID=3
Stance=+1
rebuttal=oppose
```

Figure 4.2: `abortion/A4.meta`, containing the stance value for the file depicted in fig. 4.1

Chapter 5

Implementation

Two experiment setups have been devised to make the most of both the content and structure of the `CreateDebate` data-set. Threee of the topics that are present in the `CreateDebate` data-set are abortion, gay rights and marijuana. These topics are issues. The fourth topic present in the `CreateDebate` data-set is Barack Obama, who is a person. Thus, it has been decided that the classifiers would be trained on posts about Barack Obama (the seen target) and each of the other topics in the `CreateDebate` data-set would be given a turn at being the unseen target which the classifiers would be tested upon.

5.1 Cross-Validation

Each of the classifiers has their performance evaluated in the more basic situation where the targets of training and testing are the same; both the training data and testing data are to be sourced from debates about Barack Obama. A cross-validation technique is employed where each debate about Barack Obama is given the chance to be the testing data.

In the `CreateDebate` data-set, there are 15 debates (named from A to O) about Barack Obama. Initially, A will be the testing data and 14 debates, from B to O will be the training data. Then, B will be the testing data and A, along debates C to O, will be the training data. This is done until debate O has been used as the testing data.

Seven classifiers will be evaluated; five SVM-based (one for each of five kernels; Liblinear, Linear, Polynomial, Sigmoid and Radial Basis Function) and two LSTM-based (Conditional Encoding and Bidirectional Encoding).

5.2 Seen-Unseen

The best performing SVM-based approaches and LSTM-based approaches selected from the Cross-Validation experiment described in section 5.1 will be used in this experiment.

The performance of a classifier is to be evaluated when it has been trained on posts about Barack Obama and tested on posts about the other three topics in the `CreateDebate` data-set that are not seen by the classifier.

In order to investigate why choosing certain unseen targets could alter the performance of classifiers, the bodies of posts aimed at the unseen target are concatenated into one document as is every post about Barack Obama. The document similarity between the two is then computed.

Chapter 6

Results & Discussion

6.1 Cross-Validation Results

An interesting observation were instances where there was 100% precision for the "AGAINST" stance and 100% recall for the "FAVOR" stance. For example, the Support Vector Machine with a Radial Basis Function kernel tested on posts in Obama debate C and the Bidirectional Encoding tested on posts in the Obama debate I. In these situations, the classifiers were classifying a substantial amount of the posts as "FAVOR", allowing them to correctly classify all of the "FAVOR" posts that were present in their respective debates that were used the testing data with none of these posts being mis-classified as "AGAINST", hence 100% recall.

Classifying an exceedingly large amount of posts as one stance can result in a classifier being less precise for that stance. This happened because a number of posts that were classified as "FAVOR" did not actually have that stance. The few classifications that did output the "AGAINST" stance happened to all occur on posts that were pre-labelled with the "AGAINST" stance hence 100% precision.

6.1.1 Linear SVM

The best performing approach where a Support Vector Machine is used was found to be one using a linear kernel. The classifier had an overall accuracy of 0.586069754109 across all debates. In thirteen of the iterations, the accuracy surpassed the most frequent stance in the training data. A unusual finding was the equal F-Measure for the "AGAINST" and "FAVOR" stances when tested on debate C. The precision for the "AGAINST" and the recall for the "FAVOR" stance were both $0.8\dot{3}$. On the other hand, the precision for the "FAVOR" stance and the recall for the "AGAINST" stance were both 0.625.

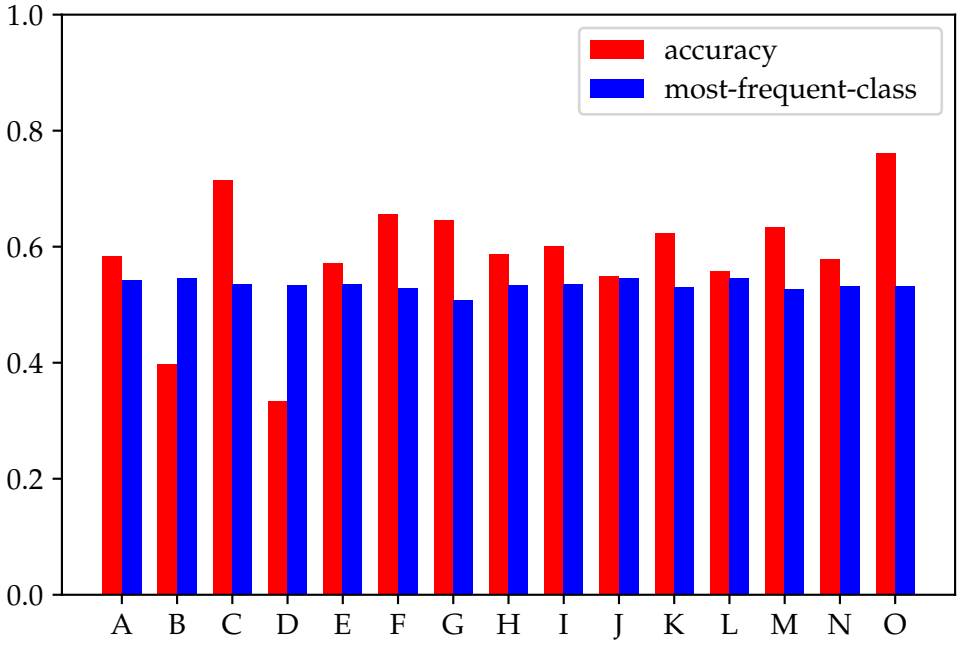


Figure 6.1: Linear SVM Accuracy versus Most Frequent Class

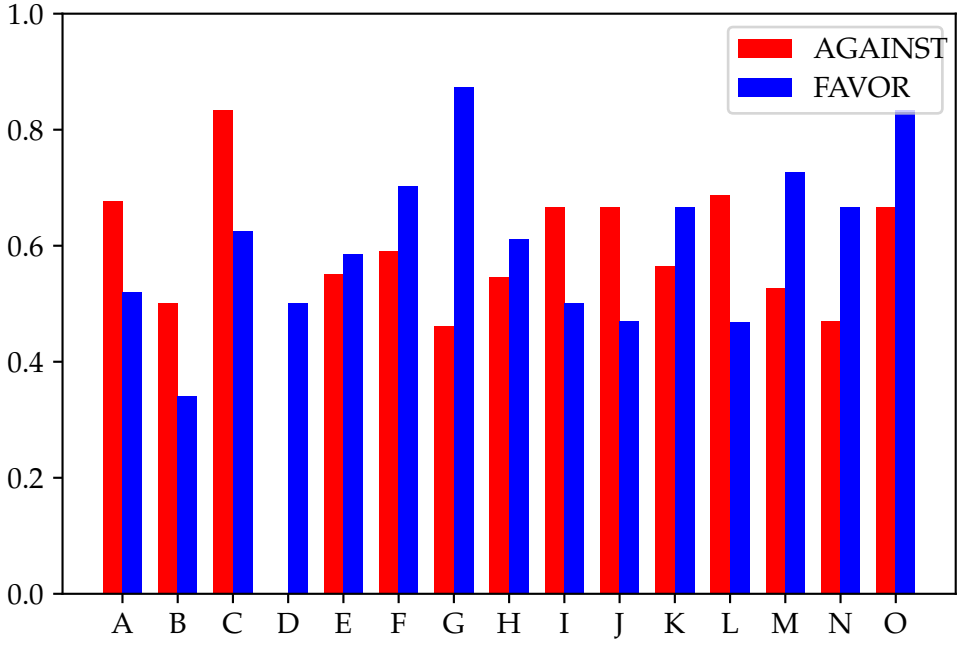


Figure 6.2: Linear SVM Precision

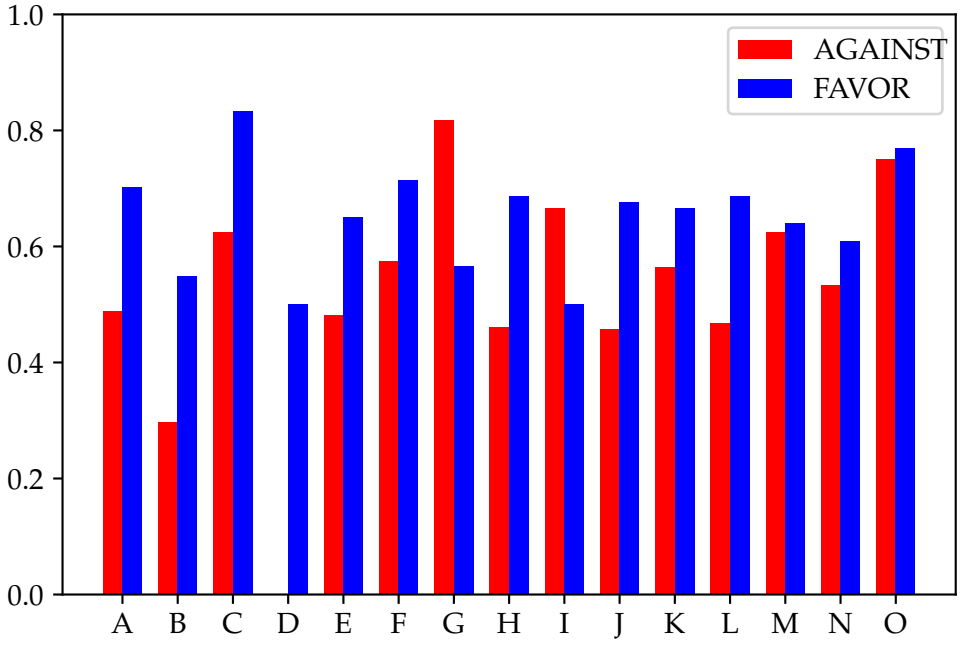


Figure 6.3: Linear SVM Recall

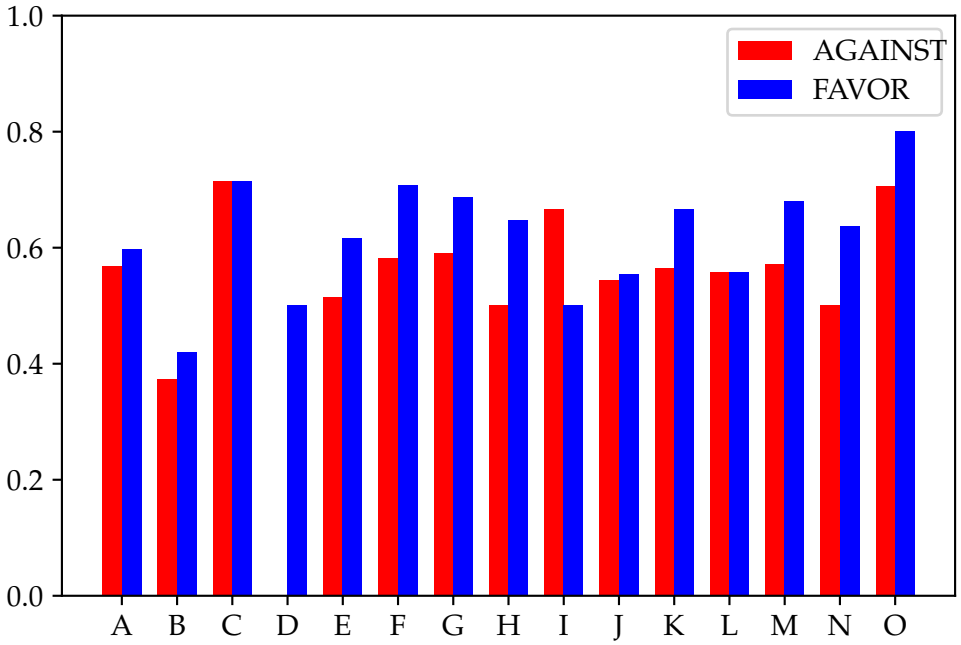


Figure 6.4: Linear SVM F-Measure

6.1.2 Radial Basis Function SVM

The second best performing Support Vector Machine was the one with a Radial Basis Function kernel, having an accuracy of 0.580186825624 across all debates and having an accuracy greater than the most frequent stance in the training data eleven times. The precision for the "AGAINST" stance and recall for the "FAVOR" stance when tested on the posts in debate C was notably high.

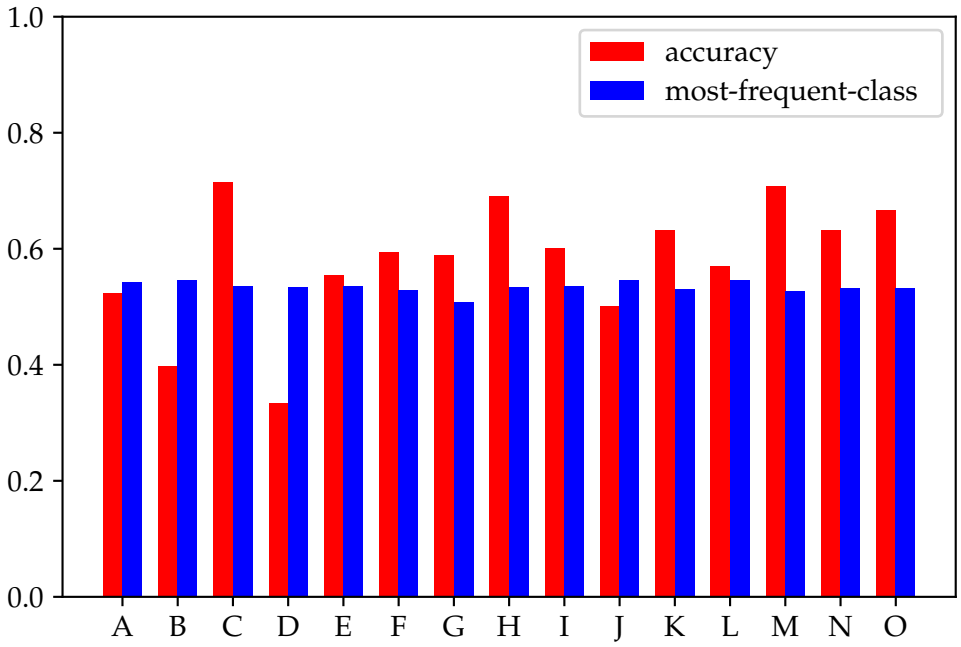


Figure 6.5: Radial Basis Function SVM Accuracy versus Most Frequent Class

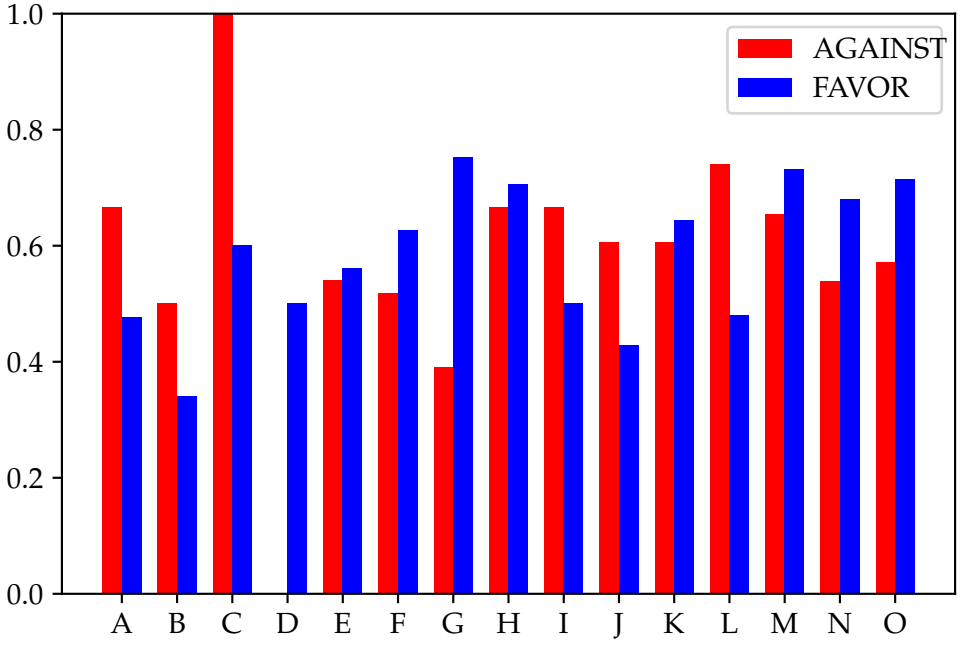


Figure 6.6: Radial Basis Function SVM Precision

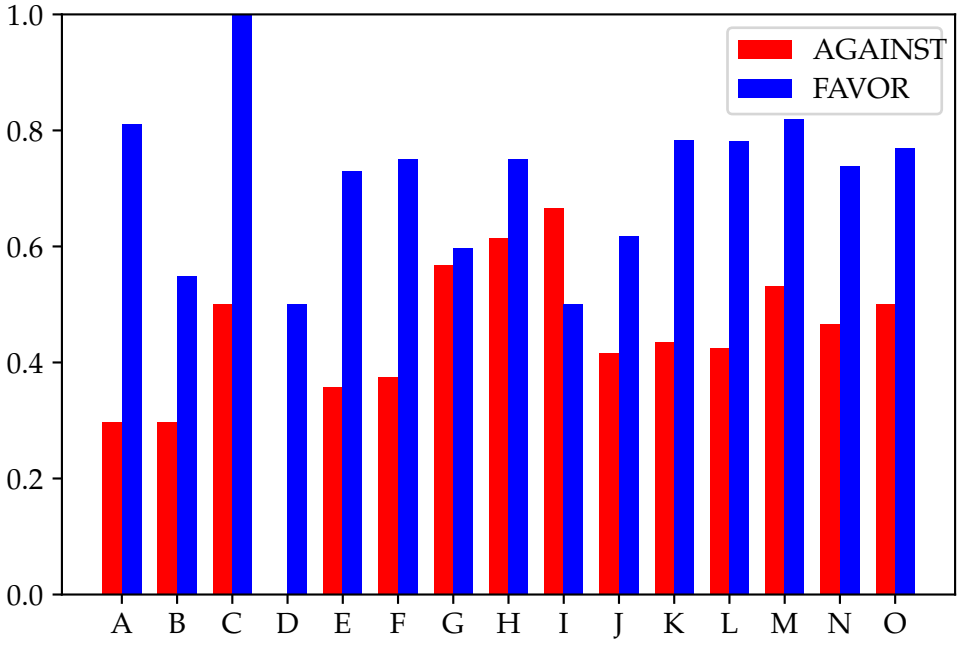


Figure 6.7: Radial Basis Function SVM Recall

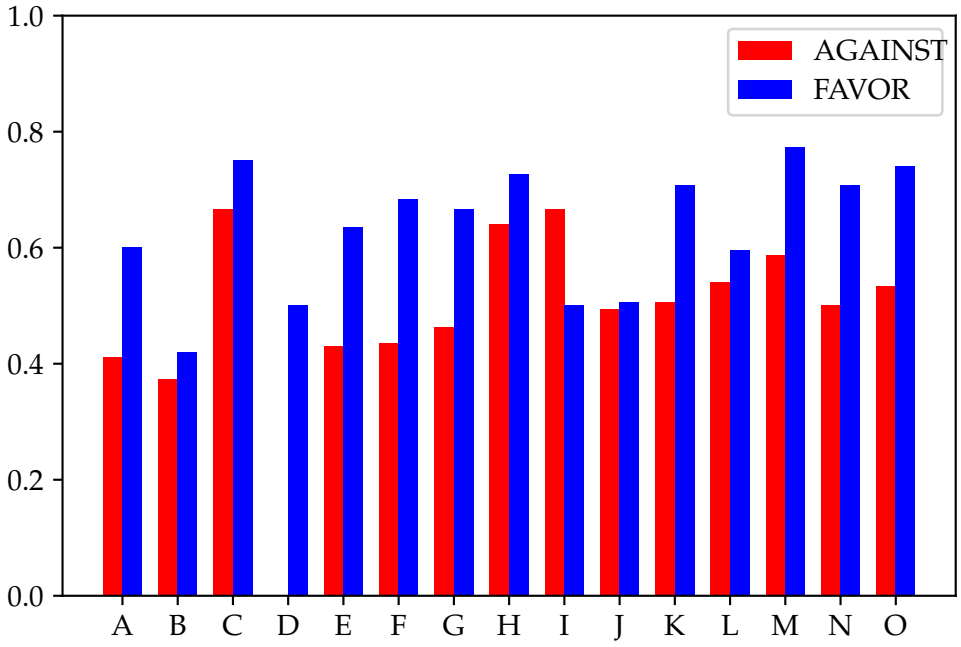


Figure 6.8: Radial Basis Function SVM F-Measure

6.1.3 Conditional Encoding

This LSTM-based approach had an accuracy of 0.545220819199 over all debates. Only the Support Vector Machine with a Polynomial kernel had a lower value at 0.543838423832. In eight of the iterations, the accuracy was greater than the proportion of the most frequent class in the training data.

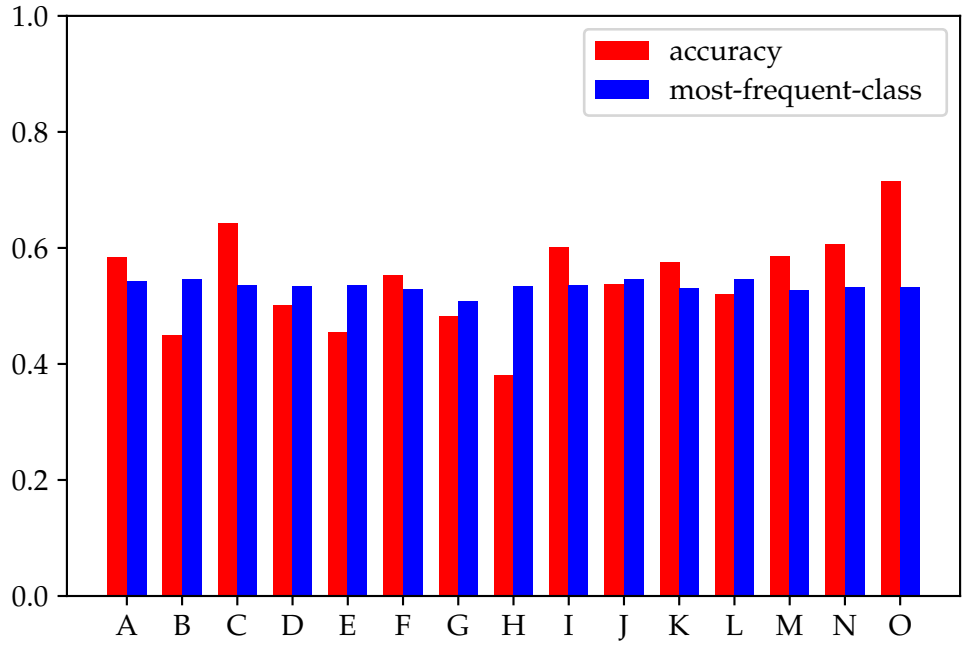


Figure 6.9: Conditional Encoding Accuracy versus Most Frequent Class

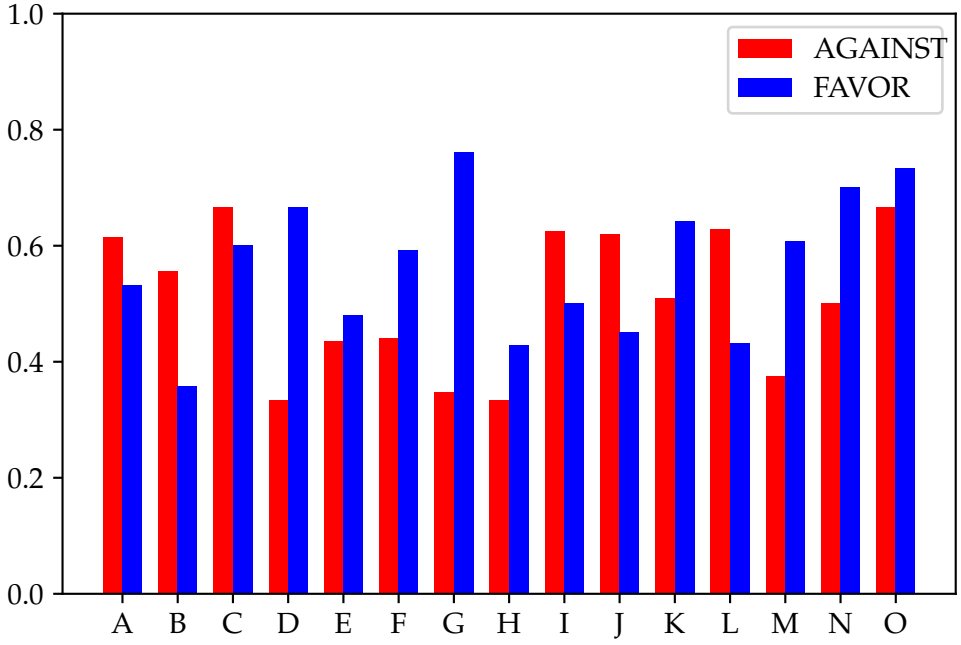


Figure 6.10: Conditional Encoding Precision

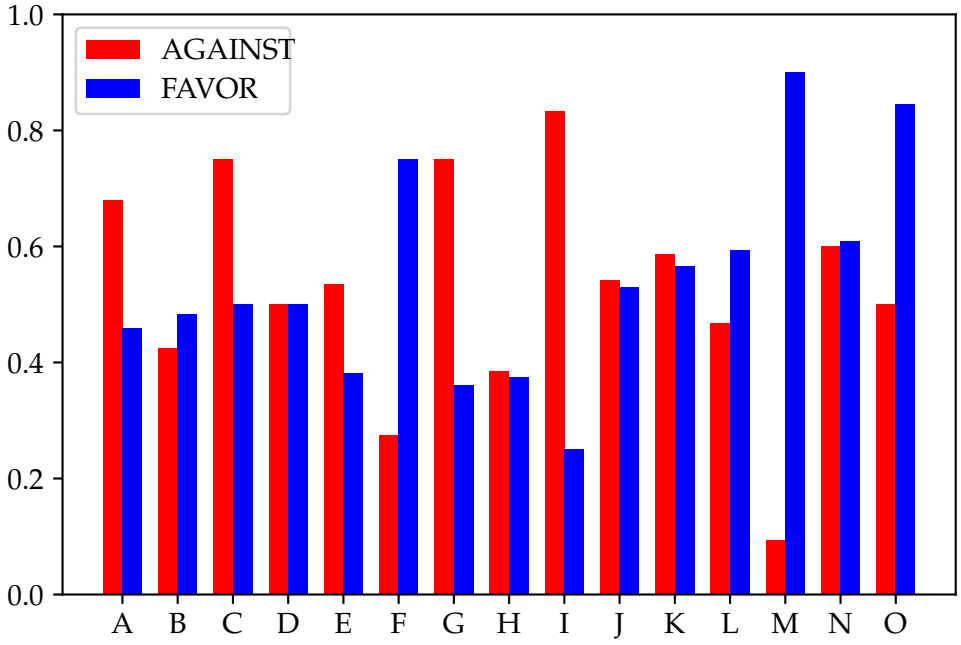


Figure 6.11: Conditional Encoding Recall

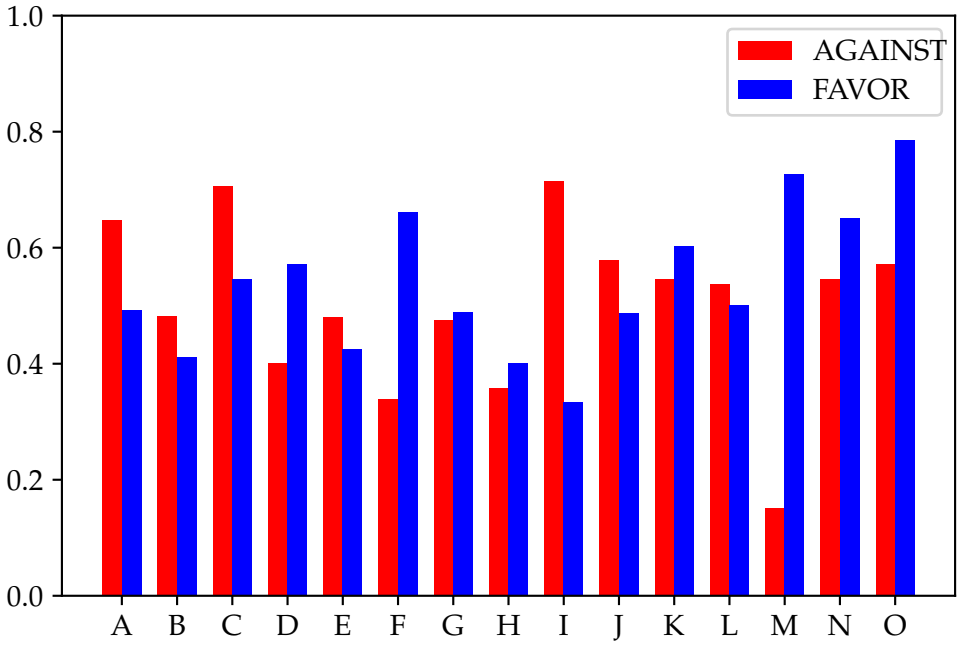


Figure 6.12: Conditional Encoding F-Measure

6.1.4 Bidirectional Encoding

The accuracy of this LSTM-based approach over all debates was 0.539134388317. Unlike Conditional Encoding, this was even less accurate than the Support Vector Machine with a Polynomial kernel. Nonetheless, just like the Conditional Encoding, in eight of the iterations, the accuracy was greater than the proportion of the most frequent class in the training data. The precision of the "AGAINST" stance and recall of the "FAVOR" stance when tested on the posts in debate I was notably high.

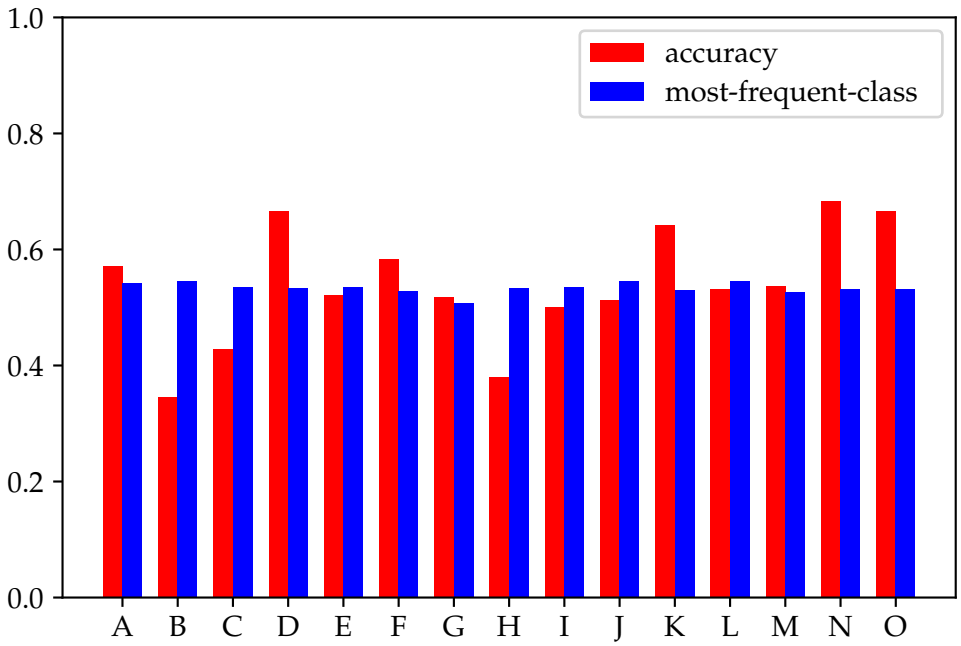


Figure 6.13: Bidirectional Encoding Accuracy versus Most Frequent Class

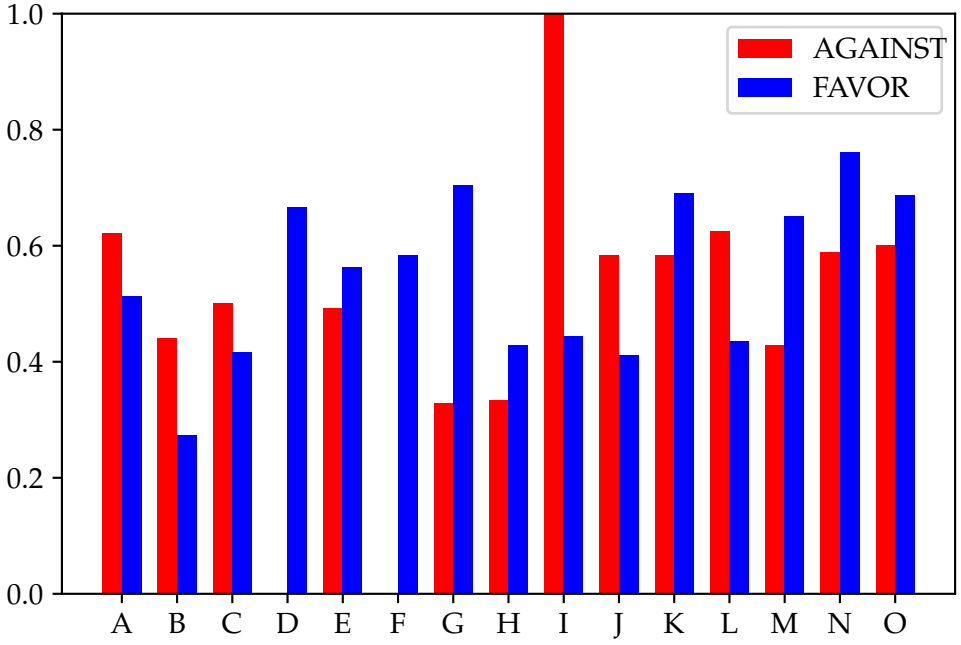


Figure 6.14: Bidirectional Encoding Precision

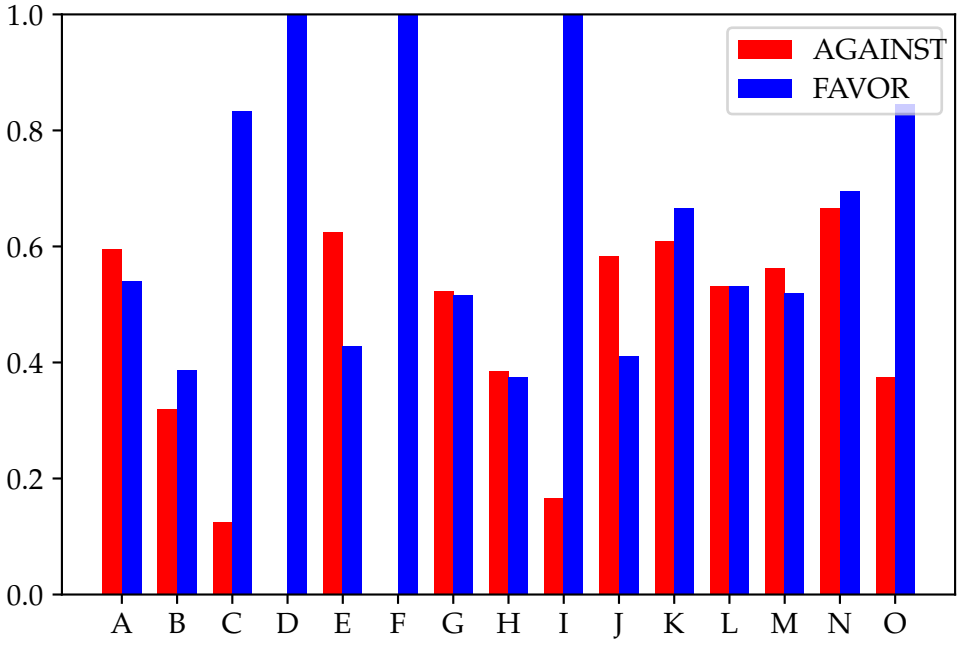


Figure 6.15: Bidirectional Encoding Recall

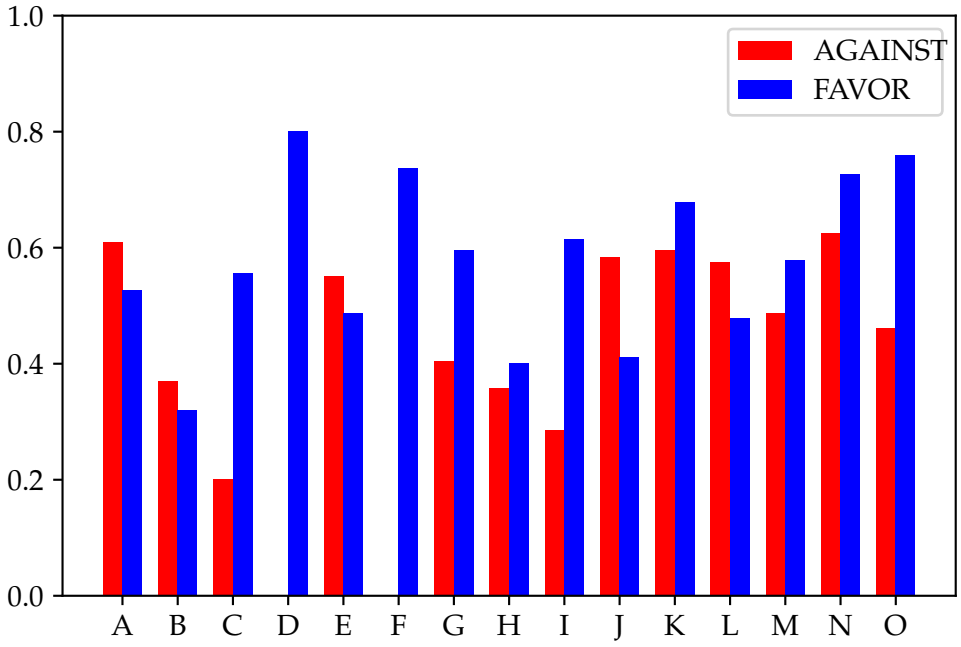


Figure 6.16: Bidirectional Encoding F-Measure

6.2 Seen-Unseen Results

The proportions of stances in both the training and testing data are split equally.

6.2.1 Abortion as the Unseen Target

The similarity between the concatenation of all debates that have been made about abortion and all debates that have been made about Barack Obama was found to be 0.77796435, making abortion the least similar topic to Barack Obama.

Linear SVM

It gives the greatest accuracy (at 0.52414605418138982) as well as precision, recall, F-Measure for the "FAVOR" stance and precision for the "AGAINST" stance.

Stance	Precision	Recall	F-Measure
AGAINST	0.5252774352651048	0.5017667844522968	0.5132530120481927
FAVOR	0.5231116121758738	0.5465253239104829	0.5345622119815668

Table 6.1: Linear SVM tested on abortion posts.

Conditional Encoding

The accuracy was found to be 0.50412249705535928, so it is the least accurate classifier when abortion is used as the unseen target. The classifier gave the greatest F-Measure and recall for the "AGAINST" stance but the smallest for "FAVOR".

Stance	Precision	Recall	F-Measure
AGAINST	0.5035175879396985	0.5901060070671378	0.5433839479392625
FAVOR	0.5049786628733998	0.41813898704358066	0.4574742268041237

Table 6.2: Conditional Encoding tested on abortion posts.

Bidirectional Encoding

Its accuracy, 0.51884570082449943, was the greatest out of the two LSTM-based approaches. It had the smallest recall and F-Measure for the "AGAINST" stance.

Stance	Precision	Recall	F-Measure
AGAINST	0.519753086419753	0.4958775029446408	0.5075346594333936
FAVOR	0.5180180180180181	0.541813898704358	0.5296488198042602

Table 6.3: Bidirectional Encoding tested on abortion posts.

6.2.2 Gay Rights as the Unseen Target

The similarity between the concatenation of all debates about gay rights and all debates about Barack Obama was found to be 0.8471235, making it the second most similar topic but the accuracy of each approach is at their lowest, lower than what they had for when the least similar topic, abortion, used as the unseen target.

Linear SVM

The accuracy was found to be 0.51503006012024044. Yet again, it was more accurate than the LSTM-based approaches. The classifier gives the smallest recall and F-Measure for the "AGAINST" stance but the greatest for the "FAVOR" stance.

Stance	Precision	Recall	F-Measure
AGAINST	0.5156576200417536	0.49498997995991983	0.5051124744376277
FAVOR	0.5144508670520231	0.5350701402805611	0.524557956777996

Table 6.4: Linear SVM tested on gay rights posts.

Conditional Encoding

The accuracy was found to be 0.48897795594168569. Again, the least accurate approach. It gave the smallest precision, recall and F-measure for the "FAVOR" stance. While it was less precise for the "AGAINST" stance than the Linear SVM, the recall and F-Measure were both greater than what the Linear SVM had.

Stance	Precision	Recall	F-Measure
AGAINST	0.4904679376083189	0.5671342685370742	0.5260223048327137
FAVOR	0.48693586698337293	0.41082164328657317	0.44565217391304346

Table 6.5: Conditional Encoding tested on gay rights posts.

Bidirectional Encoding

The accuracy was found to be 0.50300601202404804 so it is the most accurate out of the two approaches that utilise Long Short-Term Memory units. The classifier for this also gives the greatest recall and F-Measure for the "AGAINST" stance.

Stance	Precision	Recall	F-Measure
AGAINST	0.5025906735751295	0.5831663326653307	0.5398886827458256
FAVOR	0.5035799522673031	0.4228456913827655	0.4596949891067538

Table 6.6: Bidirectional Encoding tested on gay rights posts.

6.2.3 Marijuana as the Unseen Target

The similarity between the concatenation of all debates that have been made about marijuana legalisation and all debates that have been made about Barack Obama was found to be 0.85121942 making it the most similar topic. Both the Linear SVM and Bidirectional Encoding approaches give the greatest accuracy at 0.54395604362854588. Conditional Encoding gives its second smallest accuracy.

Linear SVM

It had the greatest accuracy (0.5439560439560439) as well as recall and F-measure for the "FAVOR" stance. The recall for "AGAINST" was unusually exactly a half.

Stance	Precision	Recall	F-Measure
AGAINST	0.5481927710843374	0.5	0.5229885057471264
FAVOR	0.5404040404040404	0.5879120879120879	0.5631578947368422

Table 6.7: Linear SVM tested on marijuana legalisation posts.

Conditional Encoding

The accuracy was found to be 0.49450549450549453, making it the least accurate approach for a third time. Although less precise than the Bidirectional Encoding for the "FAVOR" stance, the greater recall it gave resulted in a greater F-Measure.

Stance	Precision	Recall	F-Measure
AGAINST	0.4946808510638298	0.510989010989011	0.5027027027027027
FAVOR	0.4943181818181818	0.47802197802197804	0.48603351955307267

Table 6.8: Conditional Encoding tested on marijuana legalisation posts.

Bidirectional Encoding

The accuracy was found to be 0.5439560439560439, equal to that of the Support Vector Machine with a linear kernel. The performance of the classifier varied quite a lot between the two stances; the classifier gave the greatest recall as well as F-Measure for the "AGAINST" stance but the smallest for the "FAVOR" stance.

Stance	Precision	Recall	F-Measure
AGAINST	0.5350877192982456	0.6703296703296703	0.5951219512195122
FAVOR	0.5588235294117647	0.4175824175824176	0.47798742138364775

Table 6.9: Bidirectional Encoding tested on marijuana legalisation posts.

6.3 Further Work

The original plan for subsection 5.2 was for the unseen targets to be sourced from the `4Forums.com`data-set – another data-set previously used by Sridhar et al [3] – for this data-set had more topics than the `CreateDebate`data-set. A classifier would've been trained on debates regarding the topic of Barack Obama from the `CreateDebate`data-set but tested on the debates regarding policies of Barack Obama such as health-care, a topic that is present in the `4Forums.com`data-set

However, this came with the following limitations:

- A third stance not in the training data is present; "NONE".
- The stances that are stored were not post-level but author-level.

A technique that was considered to overcome these largely severe limitations was a hybrid classifier; a one-class classifier (OCC) would be trained on the `CreateDebate`data-set to learn posts labelled as "FAVOR" and "AGAINST", a post labelled as "NONE" from `4Forums.com`would ideally not be recognised as belonging to either "FAVOR" or "AGAINST". A post that is not classified as "NONE" would then be passed to the main classifier (which can be SVM-based or LSTM-based) which would then classify it as either "FAVOR" or "AGAINST". This would be a semi-supervised learning approach.

Having more topics would increase the number of data-points visualised in fig. 7.1, providing the sufficient data needed to conclude whether a correlation between similarity and accuracy exists oer not.

Chapter 7

Conclusions

7.1 Approaches to Stance Detection

The results in section 6.1 showed that both of the LSTM-based approaches generally performed worse when compared to the Support Vector Machine with a linear kernel with the worst of all being the Bidirectional Encoding approach. However, in section 6.2, forcing a 50/50 split for the stances in both the training data and the testing data allowed the Bidirectional Encoding approach to consistently be more accurate than the other LSTM-based approach, Conditional Encoding. Since the proportion of the most frequent stance in the training data was 0.5, the Bidirectional Encoding approach's accuracy surpassed this baseline value for all three unseen targets whereas the Conditional Encoding approach only had an accuracy greater than 0.5 when the unseen target was abortion. When marijuana legalisation was used as the unseen target, the Bidirectional Encoding approach even had an accuracy equal to that of the Support Vector Machine with a linear kernel.

Situations such as the results in section 6.1 have training data with disproportionate stances – this can be observed naturally because the stances users express on forums could be skewed by the majority opinion at time the data was collected. Here, the Conditional Encoding performs better than Bidirectional Encoding. Bidirectional Encoding can outperform Conditional Encoding when training data has equal proportions of stances, but situations like these rarely occur naturally.

7.2 Similarity & Accuracy

Figure 7.1 shows similarity between an unseen target and the seen target plotted against each classifier's accuracy for the unseen target as recorded in section 6.2. For the Linear SVM and Bidirectional Encoding approach, the most similar topic

to Barrack Obama, marijuana, gave the greatest accuraciees when it the unseen target. The least similar topic did give lower accuracies for those two classifiers as well. However, all of the classifier's lowest accuracies were when gay rights was the unseen target despite the similarity not only being large but quite close to the similarity value the marijuana topic had. The Conditional Encoding approach, defied the trend being more accurate for abortion posts than posts about marijuana.

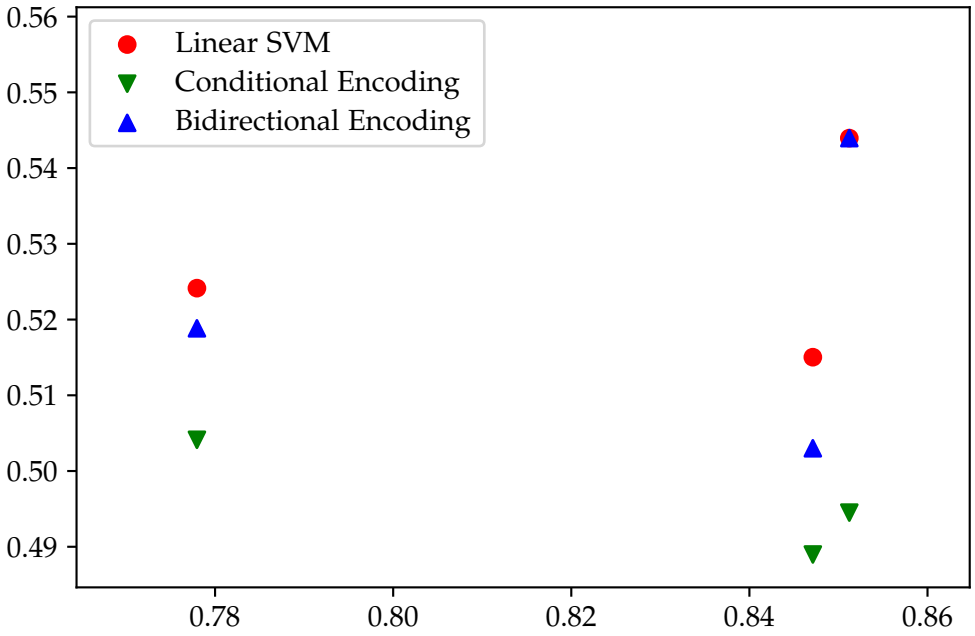


Figure 7.1: Similarity against accuracy

There is an insufficient number of data-points to conclude that there is a correlation between the following quantities:

- The similarity of two topics.
- The accuracy of classifiers when posts about one of the topics is exclusively used as training data and the other as testing data.

References

[1] I. Augenstein, T. Rocktäschel, A. Vlachos, and K. Bontcheva, "Stance detection with bidirectional conditional encoding", *CoRR*, vol. abs/1606.05464, 2016. arXiv: 1606.05464. [Online]. Available: http://arxiv.org/abs/1606.05464.

[2] P. Krejzl, B. Hourová, and J. Steinberger, "Stance detection in online discussions", *CoRR*, vol. abs/1701.00504, 2017.

[3] D. Sridhar, J. R. Foulds, B. Huang, L. Getoor, and M. A. Walker, "Joint models of disagreement and stance in online debate.", in *ACL (1)*, The Association for Computer Linguistics, 2015, pp. 116–125, ISBN: 978-1-941643-72-3. [Online]. Available: http://dblp.uni-trier.de/db/conf/acl/acl2015-1.html#SridharFHGW15.

[4] L. Shi, B. Sun, L. Kong, and Y. Zhang, "Web forum sentiment analysis based on topics", in *2009 Ninth IEEE International Conference on Computer and Information Technology*, vol. 2, Oct. 2009, pp. 148–153. DOI: 10.1109/CIT.2009.53.

[5] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, Third Edition Draft. 2017. [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/.

[6] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality", *CoRR*, vol. abs/1310.4546, 2013. arXiv: 1310.4546. [Online]. Available: http://arxiv.org/abs/1310.4546.

[7] D. Guthrie, B. Allison, W. Liu, L. Guthrie, and Y. Wilks, "A closer look at skip-gram modelling", Jan. 2006.

[8] V. N. Vapnik, "An overview of statistical learning theory", *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 988–999, Sep. 1999, ISSN: 1045-9227. DOI: 10.1109/72.788640.

[9] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up? sentiment classification using machine learning techniques", *CoRR*, vol. cs.CL/0205070, 2002. [Online]. Available: http://arxiv.org/abs/cs.CL/0205070.

[10] D. Küçük and F. Can, "Stance Detection on Tweets: An SVM-based Approach", *ArXiv e-prints*, Mar. 2018. arXiv: 1803.08910 [cs.CL].

[11] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Fourth Edition*, 4th. Academic Press, 2008, ISBN: 1597492728, 9781597492720.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[13] T. Rocktäschel, E. Grefenstette, K. M. Hermann, T. Kociský, and P. Blunsom, "Reasoning about entailment with neural attention.", *CoRR*, vol. abs/1509.06664, 2015. [Online]. Available: http://dblp.uni-trier.de/db/journals/corr/corr1509.html#RocktaschelGHKB15.

[14] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm networks", in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 4, Jul. 2005, 2047–2052 vol. 4. DOI: 10.1109/IJCNN.2005.1556215.

[15] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis", *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE*, vol. 41, no. 6, pp. 391–407, 1990.

[16] R. Gaizauskas, *Sentiment analysis: Approaches and evaluation*, https://staffwww.dcs.shef.ac.uk/people/R.Gaizauskas/campus_only/com3110/lectures/lecture_SA3.pdf, 2017.

Appendices

Cross-Validation Raw Results

Liblinear SVM

Accuracy: 0.577231639681

A

Accuracy: 0.535714285714

Metric	AGAINST	FAVOR
Precision	0.617647	0.48
Recall	0.446809	0.648649
F-measure	0.518519	0.551724
Proportion in Training Data	0.45727	0.54273

B

Accuracy: 0.435897435897

Metric	AGAINST	FAVOR
Precision	0.555556	0.372549
Recall	0.319149	0.612903
F-measure	0.405405	0.463415
Proportion in Training Data	0.454245	0.545755

C

Accuracy: 0.714285714286

Metric	AGAINST	FAVOR
Precision	0.75	0.666667
Recall	0.75	0.666667
F-measure	0.75	0.666667
Proportion in Training Data	0.46447	0.53553

D

Accuracy: 0.333333333333

Metric	AGAINST	FAVOR
Precision	0	0.5
Recall	0	0.5
F-measure	0	0.5
Proportion in Training Data	0.466803	0.533197

E

Accuracy: 0.579831932773

Metric	AGAINST	FAVOR
Precision	0.568182	0.586667
Recall	0.446429	0.698413
F-measure	0.5	0.637681
Proportion in Training Data	0.465358	0.534642

F

Accuracy: 0.65625

Metric	AGAINST	FAVOR
Precision	0.577778	0.72549
Recall	0.65	0.660714
F-measure	0.611765	0.691589
Proportion in Training Data	0.471316	0.528684

G

Accuracy: 0.624113475177

Metric	AGAINST	FAVOR
Precision	0.441558	0.84375
Recall	0.772727	0.556701
F-measure	0.561983	0.670807
Proportion in Training Data	0.491706	0.508294

H

Accuracy: 0.551724137931

Metric	AGAINST	FAVOR
Precision	0.5	0.578947
Recall	0.384615	0.6875
F-measure	0.434783	0.628571
Proportion in Training Data	0.466527	0.533473

I

Accuracy: 0.6

Metric	AGAINST	FAVOR
Precision	0.666667	0.5
Recall	0.666667	0.5
F-measure	0.666667	0.5
Proportion in Training Data	0.464615	0.535385

J

Accuracy: 0.512195121951

Metric	AGAINST	FAVOR
Precision	0.605263	0.431818
Recall	0.479167	0.558824
F-measure	0.534884	0.487179
Proportion in Training Data	0.45515	0.54485

K

Accuracy: 0.603773584906

Metric	AGAINST	FAVOR
Precision	0.541667	0.655172
Recall	0.565217	0.633333
F-measure	0.553191	0.644068
Proportion in Training Data	0.469852	0.530148

L

Accuracy: 0.544303797468

Metric	AGAINST	FAVOR
Precision	0.677419	0.458333
Recall	0.446809	0.6875
F-measure	0.538462	0.55
Proportion in Training Data	0.454746	0.545254

M

Accuracy: 0.69512195122

Metric	AGAINST	FAVOR
Precision	0.594595	0.777778
Recall	0.6875	0.7
F-measure	0.637681	0.736842
Proportion in Training Data	0.472868	0.527132

N

Accuracy: 0.605263157895

Metric	AGAINST	FAVOR
Precision	0.5	0.681818
Recall	0.533333	0.652174
F-measure	0.516129	0.666667
Proportion in Training Data	0.468849	0.531151

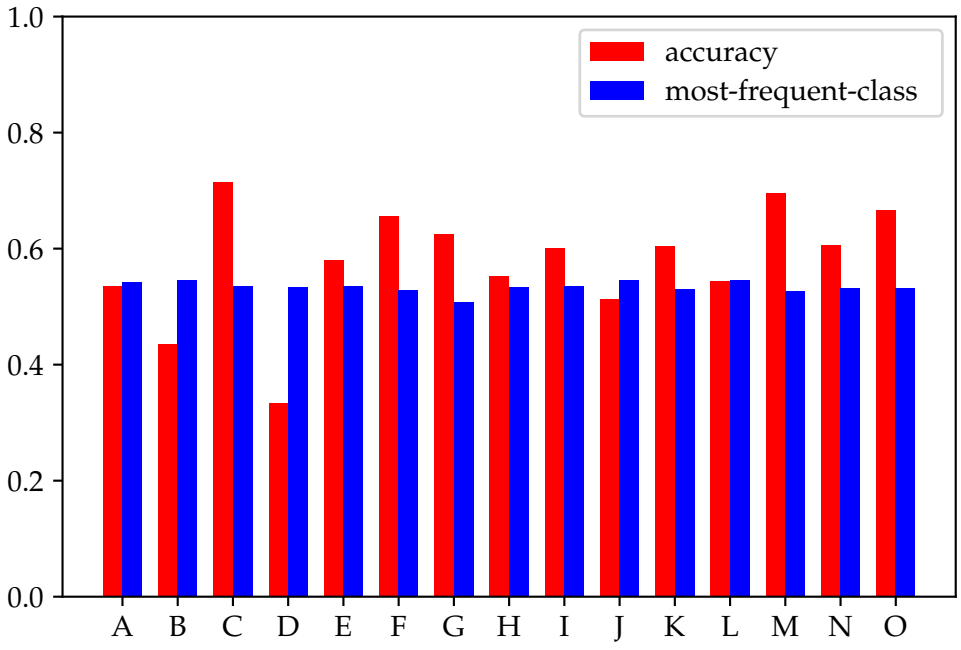
O

Accuracy: 0.666666666667

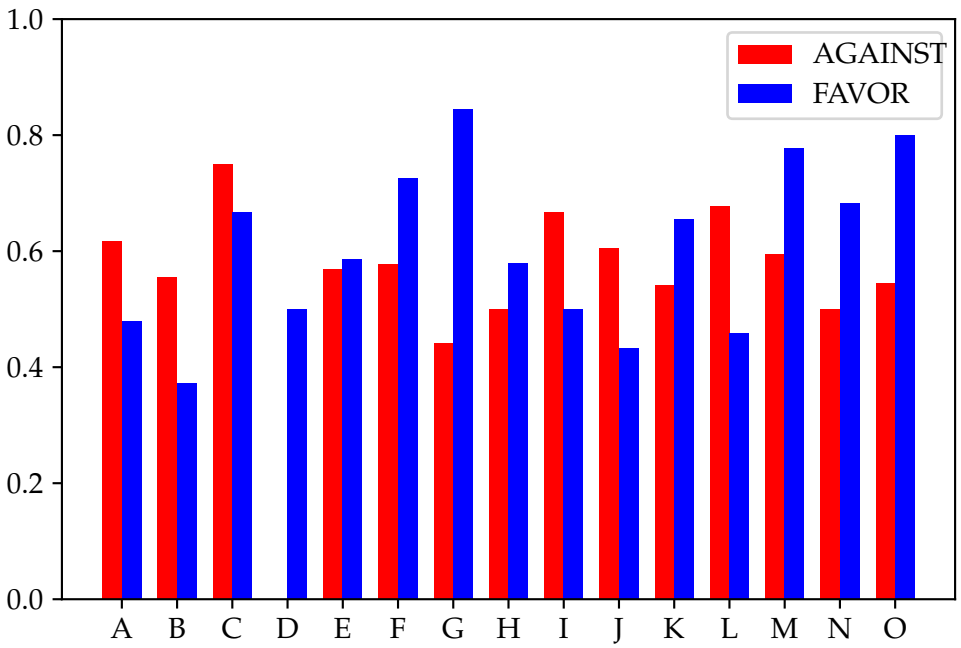
Metric	AGAINST	FAVOR
Precision	0.545455	0.8
Recall	0.75	0.615385
F-measure	0.631579	0.695652
Proportion in Training Data	0.467842	0.532158

Number of times accuracy was greater than proportion of most frequent stance in training data: 10

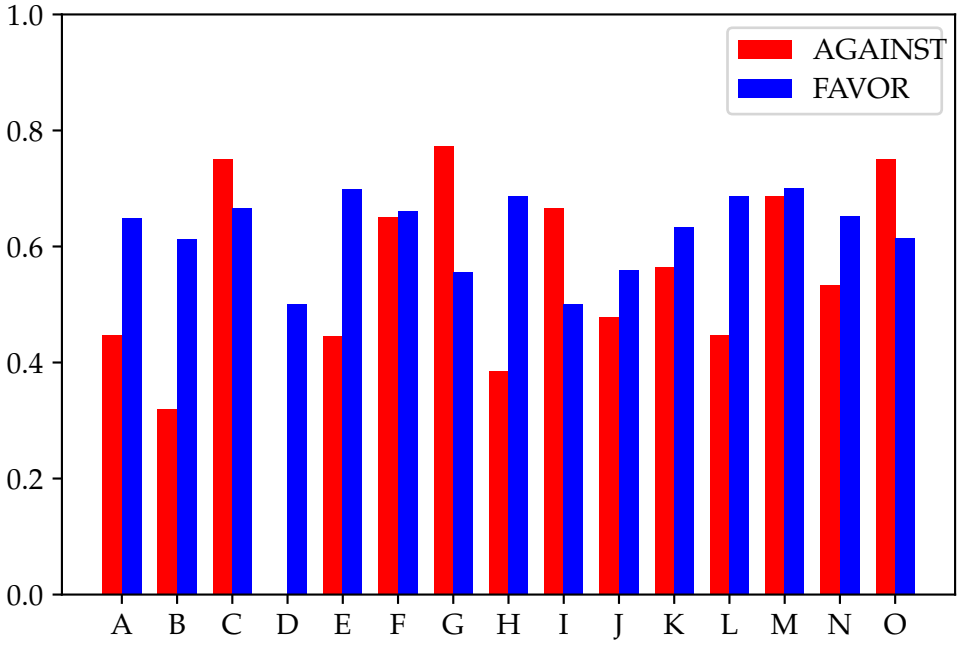
Accuracy



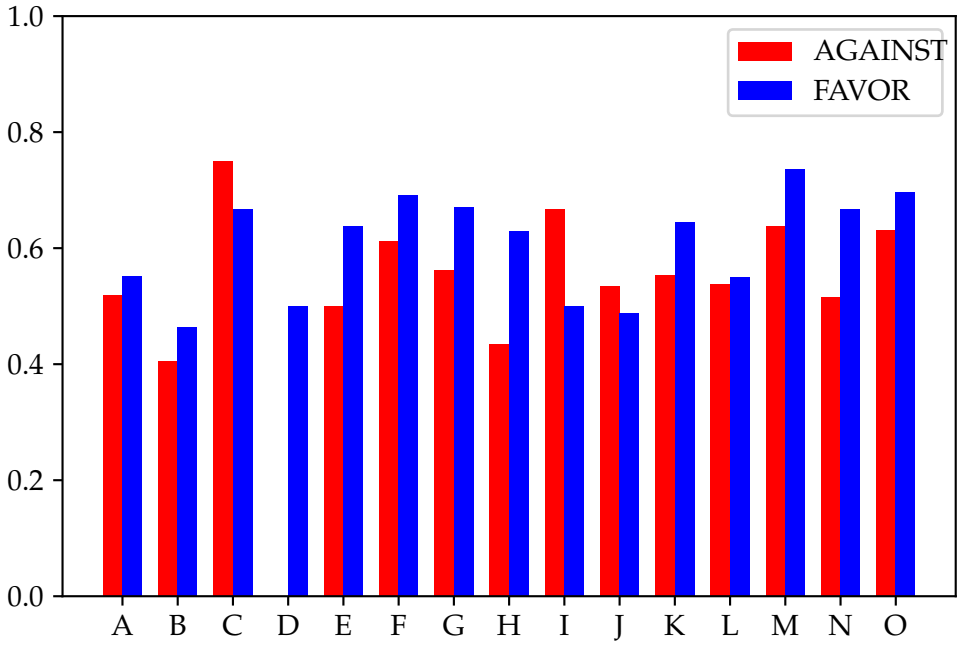
Precision



Recall



F-Measure



Linear SVM

Accuracy: 0.586069754109

A

Accuracy: 0.583333333333

Metric	AGAINST	FAVOR
Precision	0.676471	0.52
Recall	0.489362	0.702703
F-measure	0.567901	0.597701
Proportion in Training Data	0.45727	0.54273

B

Accuracy: 0.397435897436

Metric	AGAINST	FAVOR
Precision	0.5	0.34
Recall	0.297872	0.548387
F-measure	0.373333	0.419753
Proportion in Training Data	0.454245	0.545755

C

Accuracy: 0.714285714286

Metric	AGAINST	FAVOR
Precision	0.833333	0.625
Recall	0.625	0.833333
F-measure	0.714286	0.714286
Proportion in Training Data	0.46447	0.53553

D

Accuracy: 0.333333333333

Metric	AGAINST	FAVOR
Precision	0	0.5
Recall	0	0.5
F-measure	0	0.5
Proportion in Training Data	0.466803	0.533197

E

Accuracy: 0.571428571429

Metric	AGAINST	FAVOR
Precision	0.55102	0.585714
Recall	0.482143	0.650794
F-measure	0.514286	0.616541
Proportion in Training Data	0.465358	0.534642

F

Accuracy: 0.65625

Metric	AGAINST	FAVOR
Precision	0.589744	0.701754
Recall	0.575	0.714286
F-measure	0.582278	0.707965
Proportion in Training Data	0.471316	0.528684

G

Accuracy: 0.645390070922

Metric	AGAINST	FAVOR
Precision	0.461538	0.873016
Recall	0.818182	0.56701
F-measure	0.590164	0.6875
Proportion in Training Data	0.491706	0.508294

H

Accuracy: 0.586206896552

Metric	AGAINST	FAVOR
Precision	0.545455	0.611111
Recall	0.461538	0.6875
F-measure	0.5	0.647059
Proportion in Training Data	0.466527	0.533473

I

Accuracy: 0.6

Metric	AGAINST	FAVOR
Precision	0.666667	0.5
Recall	0.666667	0.5
F-measure	0.666667	0.5
Proportion in Training Data	0.464615	0.535385

J

Accuracy: 0.548780487805

Metric	AGAINST	FAVOR
Precision	0.666667	0.469388
Recall	0.458333	0.676471
F-measure	0.54321	0.554217
Proportion in Training Data	0.45515	0.54485

K

Accuracy: 0.622641509434

Metric	AGAINST	FAVOR
Precision	0.565217	0.666667
Recall	0.565217	0.666667
F-measure	0.565217	0.666667
Proportion in Training Data	0.469852	0.530148

L

Accuracy: 0.556962025316

Metric	AGAINST	FAVOR
Precision	0.6875	0.468085
Recall	0.468085	0.6875
F-measure	0.556962	0.556962
Proportion in Training Data	0.454746	0.545254

M

Accuracy: 0.634146341463

Metric	AGAINST	FAVOR
Precision	0.526316	0.727273
Recall	0.625	0.64
F-measure	0.571429	0.680851
Proportion in Training Data	0.472868	0.527132

N

Accuracy: 0.578947368421

Metric	AGAINST	FAVOR
Precision	0.470588	0.666667
Recall	0.533333	0.608696
F-measure	0.5	0.636364
Proportion in Training Data	0.468849	0.531151

O

Accuracy: 0.761904761905

Metric	AGAINST	FAVOR
Precision	0.666667	0.833333
Recall	0.75	0.769231
F-measure	0.705882	0.8
Proportion in Training Data	0.467842	0.532158

Number of times accuracy was greater than proportion of most frequent stance in training data: 13

Polynomial SVM

Accuracy: 0.543838423832

A

Accuracy: 0.440476190476

Metric	AGAINST	FAVOR
Precision	0	0.440476
Recall	0	1
F-measure	0	0.61157
Proportion in Training Data	0.45727	0.54273

B

Accuracy: 0.423076923077

Metric	AGAINST	FAVOR
Precision	0.666667	0.402778
Recall	0.0851064	0.935484
F-measure	0.150943	0.563107
Proportion in Training Data	0.454245	0.545755

C

Accuracy: 0.5

Metric	AGAINST	FAVOR
Precision	1	0.461538
Recall	0.125	1
F-measure	0.222222	0.631579
Proportion in Training Data	0.46447	0.53553

D

Accuracy: 0.666666666667

Metric	AGAINST	FAVOR
Precision	0	0.666667
Recall	0	1
F-measure	0	0.8
Proportion in Training Data	0.466803	0.533197

E

Accuracy: 0.546218487395

Metric	AGAINST	FAVOR
Precision	0.75	0.53913
Recall	0.0535714	0.984127
F-measure	0.1	0.696629
Proportion in Training Data	0.465358	0.534642

F

Accuracy: 0.604166666667

Metric	AGAINST	FAVOR
Precision	0.625	0.602273
Recall	0.125	0.946429
F-measure	0.208333	0.736111
Proportion in Training Data	0.471316	0.528684

G

Accuracy: 0.666666666667

Metric	AGAINST	FAVOR
Precision	0	0.681159
Recall	0	0.969072
F-measure	0	0.8
Proportion in Training Data	0.491706	0.508294

H

Accuracy: 0.51724137931

Metric	AGAINST	FAVOR
Precision	0	0.535714
Recall	0	0.9375
F-measure	0	0.681818
Proportion in Training Data	0.466527	0.533473

I

Accuracy: 0.5

Metric	AGAINST	FAVOR
Precision	0.666667	0.428571
Recall	0.333333	0.75
F-measure	0.444444	0.545455
Proportion in Training Data	0.464615	0.535385

J

Accuracy: 0.414634146341

Metric	AGAINST	FAVOR
Precision	0	0.414634
Recall	0	1
F-measure	0	0.586207
Proportion in Training Data	0.45515	0.54485

K

Accuracy: 0.575471698113

Metric	AGAINST	FAVOR
Precision	0.666667	0.572816
Recall	0.0434783	0.983333
F-measure	0.0816327	0.723926
Proportion in Training Data	0.469852	0.530148

L

Accuracy: 0.430379746835

Metric	AGAINST	FAVOR
Precision	0.75	0.413333
Recall	0.0638298	0.96875
F-measure	0.117647	0.579439
Proportion in Training Data	0.454746	0.545254

M

Accuracy: 0.621951219512

Metric	AGAINST	FAVOR
Precision	0.666667	0.620253
Recall	0.0625	0.98
F-measure	0.114286	0.75969
Proportion in Training Data	0.472868	0.527132

N

Accuracy: 0.631578947368

Metric	AGAINST	FAVOR
Precision	0.666667	0.628571
Recall	0.133333	0.956522
F-measure	0.222222	0.758621
Proportion in Training Data	0.468849	0.531151

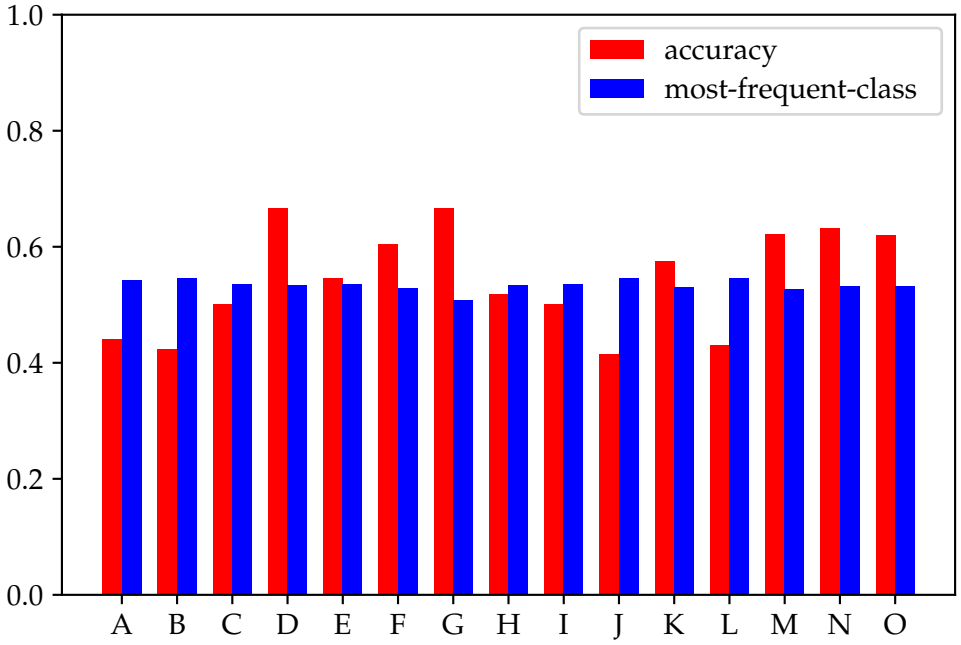
O

Accuracy: 0.619047619048

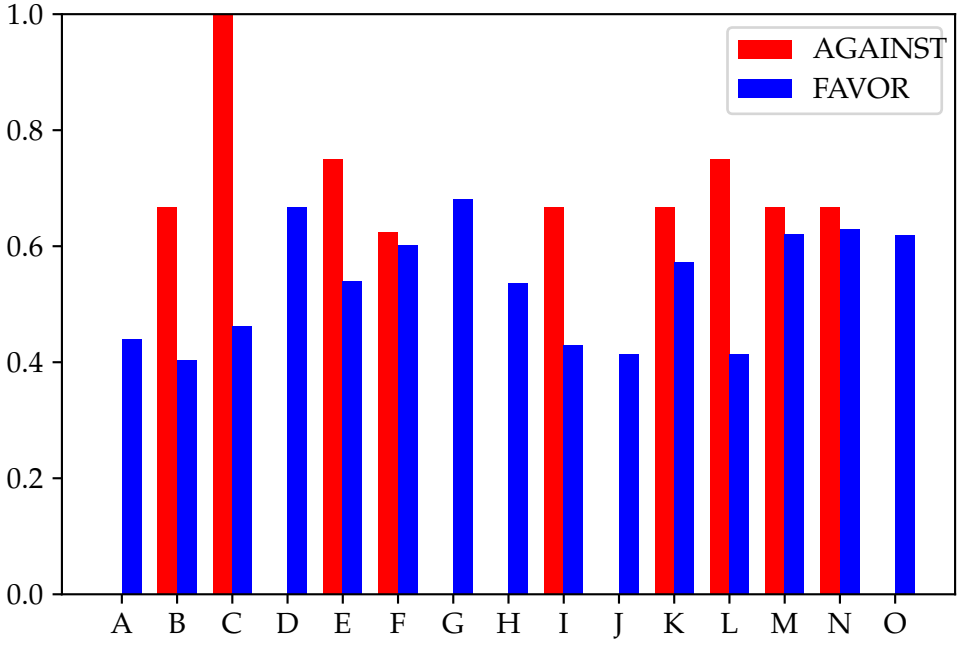
Metric	AGAINST	FAVOR
Precision	0	0.619048
Recall	0	1
F-measure	0	0.764706
Proportion in Training Data	0.467842	0.532158

Number of times accuracy was greater than proportion of most frequent stance in training data: 8

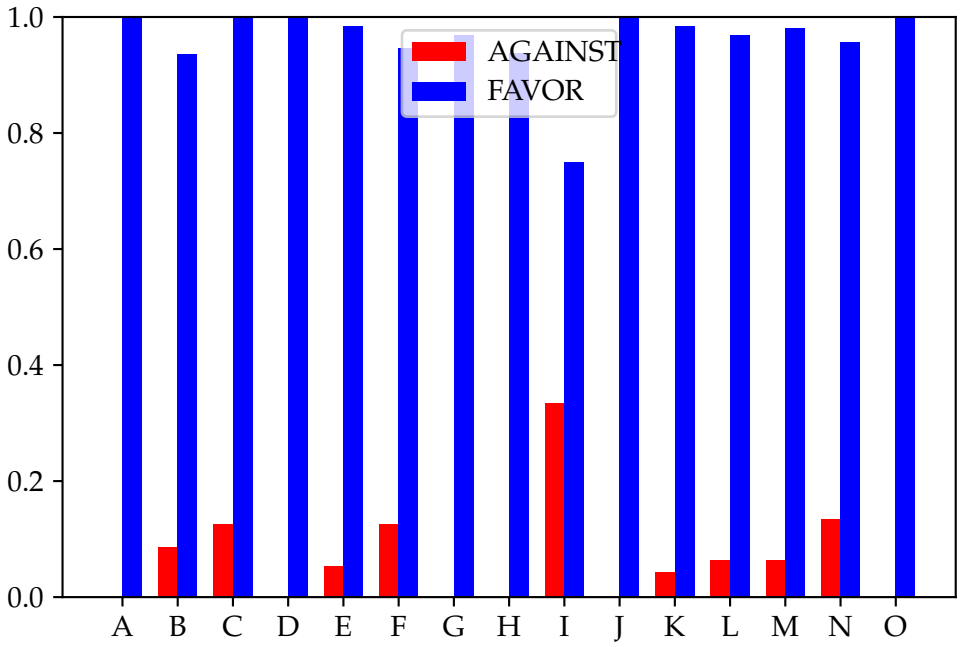
Accuracy



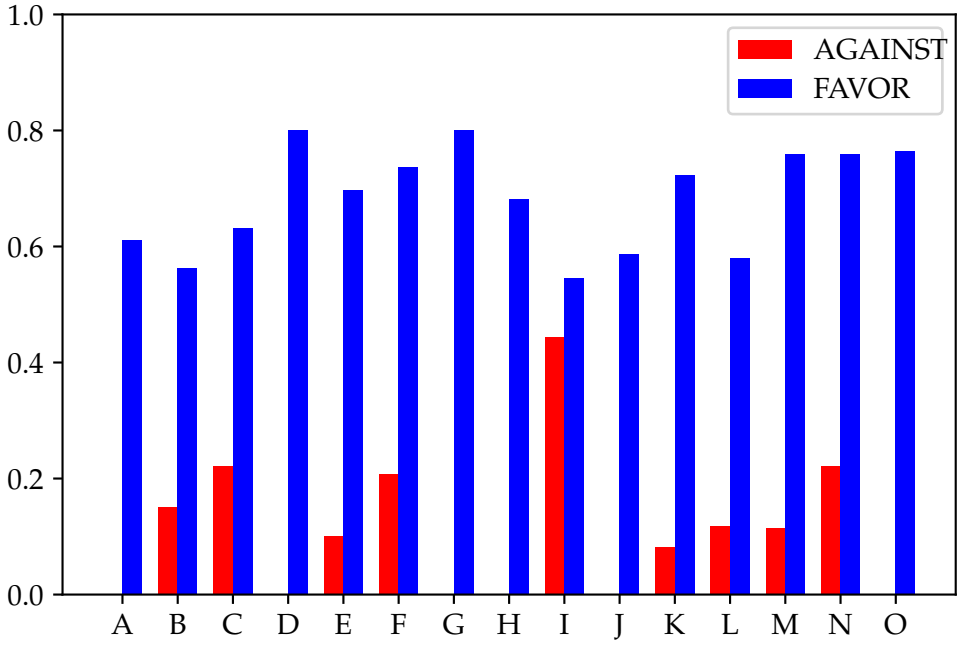
Precision



Recall



F-Measure



Sigmoid SVM

Accuracy: 0.569190274189

A

Accuracy: 0.547619047619

Metric	AGAINST	FAVOR
Precision	0.645161	0.490566
Recall	0.425532	0.702703
F-measure	0.512821	0.577778
Proportion in Training Data	0.45727	0.54273

B

Accuracy: 0.358974358974

Metric	AGAINST	FAVOR
Precision	0.451613	0.297872
Recall	0.297872	0.451613
F-measure	0.358974	0.358974
Proportion in Training Data	0.454245	0.545755

C

Accuracy: 0.714285714286

Metric	AGAINST	FAVOR
Precision	0.833333	0.625
Recall	0.625	0.833333
F-measure	0.714286	0.714286
Proportion in Training Data	0.46447	0.53553

D

Accuracy: 0.5

Metric	AGAINST	FAVOR
Precision	0.333333	0.666667
Recall	0.5	0.5
F-measure	0.4	0.571429
Proportion in Training Data	0.466803	0.533197

E

Accuracy: 0.563025210084

Metric	AGAINST	FAVOR
Precision	0.537037	0.584615
Recall	0.517857	0.603175
F-measure	0.527273	0.59375
Proportion in Training Data	0.465358	0.534642

F

Accuracy: 0.625

Metric	AGAINST	FAVOR
Precision	0.54	0.717391
Recall	0.675	0.589286
F-measure	0.6	0.647059
Proportion in Training Data	0.471316	0.528684

G

Accuracy: 0.659574468085

Metric	AGAINST	FAVOR
Precision	0.47561	0.915254
Recall	0.886364	0.556701
F-measure	0.619048	0.692308
Proportion in Training Data	0.491706	0.508294

H

Accuracy: 0.586206896552

Metric	AGAINST	FAVOR
Precision	0.538462	0.625
Recall	0.538462	0.625
F-measure	0.538462	0.625
Proportion in Training Data	0.466527	0.533473

I

Accuracy: 0.5

Metric	AGAINST	FAVOR
Precision	0.6	0.4
Recall	0.5	0.5
F-measure	0.545455	0.444444
Proportion in Training Data	0.464615	0.535385

J

Accuracy: 0.573170731707

Metric	AGAINST	FAVOR
Precision	0.658537	0.487805
Recall	0.5625	0.588235
F-measure	0.606742	0.533333
Proportion in Training Data	0.45515	0.54485

K

Accuracy: 0.61320754717

Metric	AGAINST	FAVOR
Precision	0.55102	0.666667
Recall	0.586957	0.633333
F-measure	0.568421	0.649573
Proportion in Training Data	0.469852	0.530148

L

Accuracy: 0.53164556962

Metric	AGAINST	FAVOR
Precision	0.666667	0.44898
Recall	0.425532	0.6875
F-measure	0.519481	0.54321
Proportion in Training Data	0.454746	0.545254

M

Accuracy: 0.609756097561

Metric	AGAINST	FAVOR
Precision	0.5	0.695652
Recall	0.5625	0.64
F-measure	0.529412	0.666667
Proportion in Training Data	0.472868	0.527132

N

Accuracy: 0.631578947368

Metric	AGAINST	FAVOR
Precision	0.526316	0.736842
Recall	0.666667	0.608696
F-measure	0.588235	0.666667
Proportion in Training Data	0.468849	0.531151

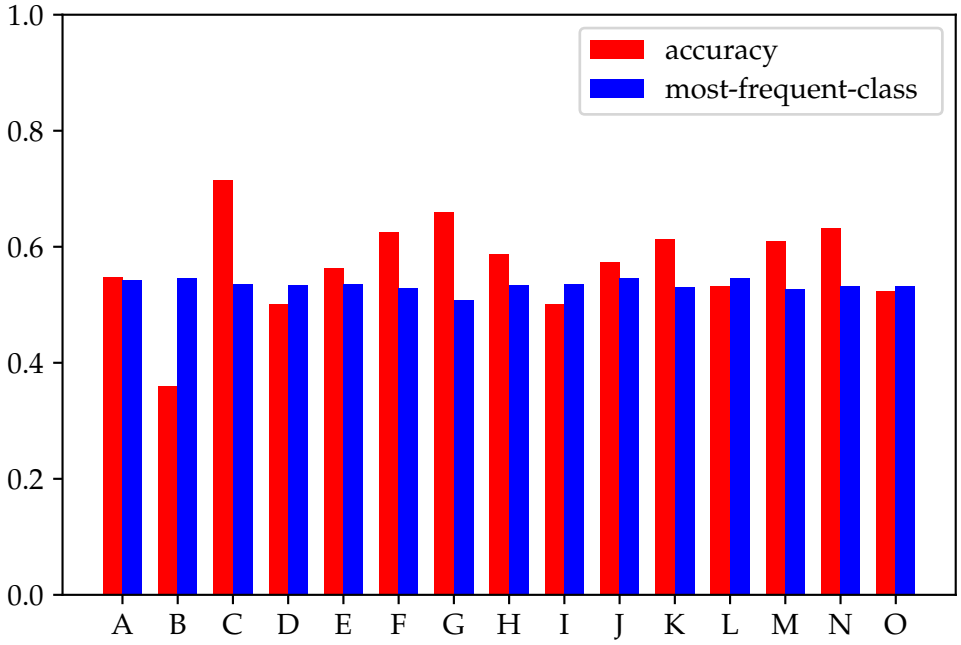
O

Accuracy: 0.52380952381

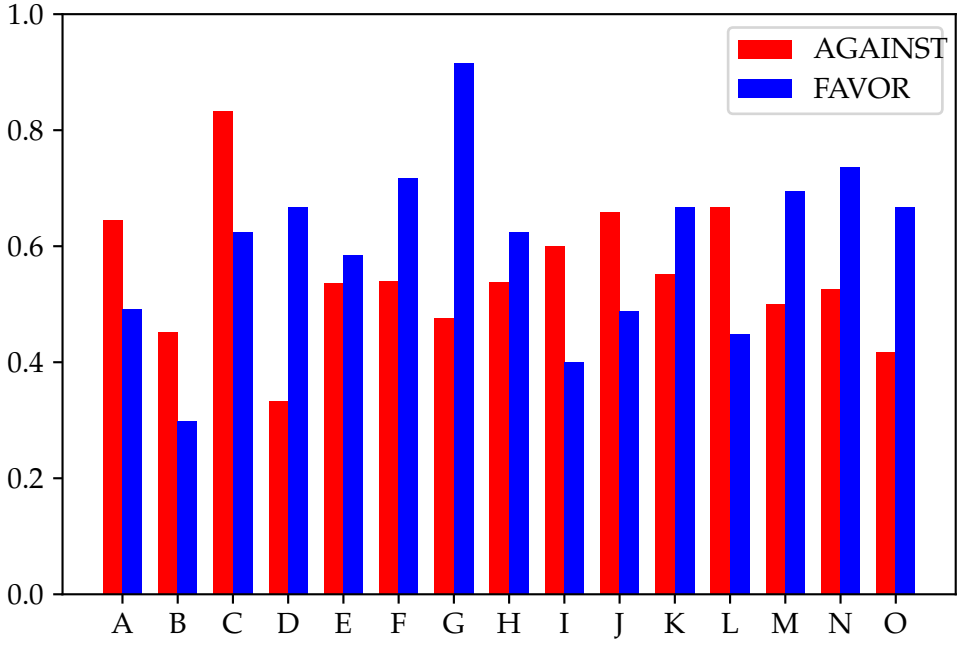
Metric	AGAINST	FAVOR
Precision	0.416667	0.666667
Recall	0.625	0.461538
F-measure	0.5	0.545455
Proportion in Training Data	0.467842	0.532158

Number of times accuracy was greater than proportion of most frequent stance in training data: 10

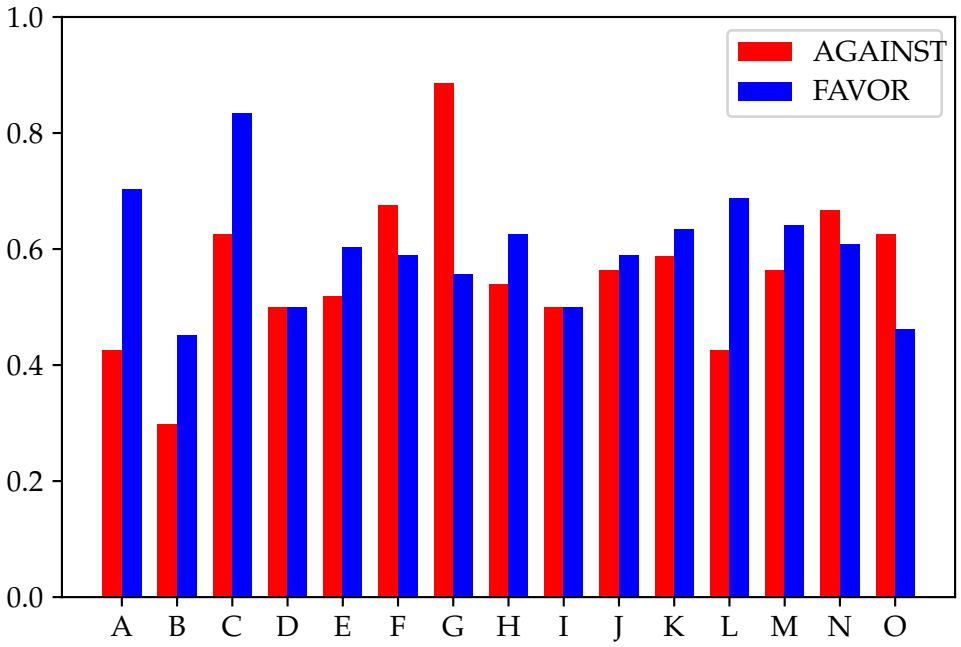
Accuracy



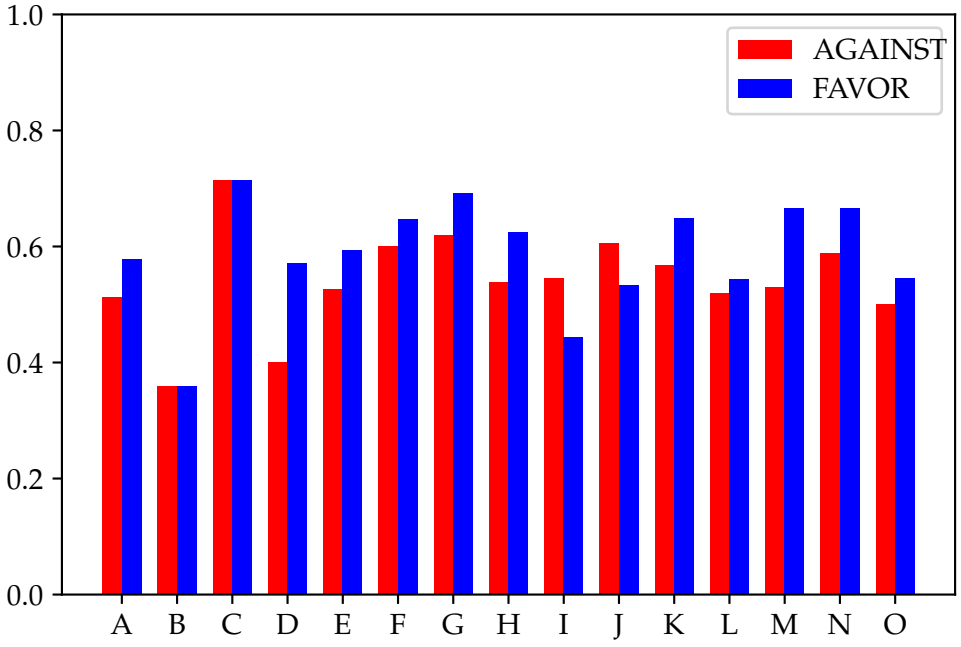
Precision



Recall



F-Measure



Radial Basis Function SVM

Accuracy: 0.580186825624

A

Accuracy: 0.52380952381

Metric	AGAINST	FAVOR
Precision	0.666667	0.47619
Recall	0.297872	0.810811
F-measure	0.411765	0.6
Proportion in Training Data	0.45727	0.54273

B

Accuracy: 0.397435897436

Metric	AGAINST	FAVOR
Precision	0.5	0.34
Recall	0.297872	0.548387
F-measure	0.373333	0.419753
Proportion in Training Data	0.454245	0.545755

C

Accuracy: 0.714285714286

Metric	AGAINST	FAVOR
Precision	1	0.6
Recall	0.5	1
F-measure	0.666667	0.75
Proportion in Training Data	0.46447	0.53553

D

Accuracy: 0.333333333333

Metric	AGAINST	FAVOR
Precision	0	0.5
Recall	0	0.5
F-measure	0	0.5
Proportion in Training Data	0.466803	0.533197

E

Accuracy: 0.554621848739

Metric	AGAINST	FAVOR
Precision	0.540541	0.560976
Recall	0.357143	0.730159
F-measure	0.430108	0.634483
Proportion in Training Data	0.465358	0.534642

F

Accuracy: 0.59375

Metric	AGAINST	FAVOR
Precision	0.517241	0.626866
Recall	0.375	0.75
F-measure	0.434783	0.682927
Proportion in Training Data	0.471316	0.528684

G

Accuracy: 0.58865248227

Metric	AGAINST	FAVOR
Precision	0.390625	0.753247
Recall	0.568182	0.597938
F-measure	0.462963	0.666667
Proportion in Training Data	0.491706	0.508294

H

Accuracy: 0.689655172414

Metric	AGAINST	FAVOR
Precision	0.666667	0.705882
Recall	0.615385	0.75
F-measure	0.64	0.727273
Proportion in Training Data	0.466527	0.533473

I

Accuracy: 0.6

Metric	AGAINST	FAVOR
Precision	0.666667	0.5
Recall	0.666667	0.5
F-measure	0.666667	0.5
Proportion in Training Data	0.464615	0.535385

J

Accuracy: 0.5

Metric	AGAINST	FAVOR
Precision	0.606061	0.428571
Recall	0.416667	0.617647
F-measure	0.493827	0.506024
Proportion in Training Data	0.45515	0.54485

K

Accuracy: 0.632075471698

Metric	AGAINST	FAVOR
Precision	0.606061	0.643836
Recall	0.434783	0.783333
F-measure	0.506329	0.706767
Proportion in Training Data	0.469852	0.530148

L

Accuracy: 0.569620253165

Metric	AGAINST	FAVOR
Precision	0.740741	0.480769
Recall	0.425532	0.78125
F-measure	0.540541	0.595238
Proportion in Training Data	0.454746	0.545254

M

Accuracy: 0.707317073171

Metric	AGAINST	FAVOR
Precision	0.653846	0.732143
Recall	0.53125	0.82
F-measure	0.586207	0.773585
Proportion in Training Data	0.472868	0.527132

N

Accuracy: 0.631578947368

Metric	AGAINST	FAVOR
Precision	0.538462	0.68
Recall	0.466667	0.73913
F-measure	0.5	0.708333
Proportion in Training Data	0.468849	0.531151

O

Accuracy: 0.666666666667

Metric	AGAINST	FAVOR
Precision	0.571429	0.714286
Recall	0.5	0.769231
F-measure	0.533333	0.740741
Proportion in Training Data	0.467842	0.532158

Conditional Encoding

Accuracy: 0.545220819199

A

Accuracy: 0.583333330495

Metric	AGAINST	FAVOR
Precision	0.615385	0.53125
Recall	0.680851	0.459459
F-measure	0.646465	0.492754
Proportion in Training Data	0.45727	0.54273

B

Accuracy: 0.448717951775

Metric	AGAINST	FAVOR
Precision	0.555556	0.357143
Recall	0.425532	0.483871
F-measure	0.481928	0.410959
Proportion in Training Data	0.454245	0.545755

C

Accuracy: 0.642857134342

Metric	AGAINST	FAVOR
Precision	0.666667	0.6
Recall	0.75	0.5
F-measure	0.705882	0.545455
Proportion in Training Data	0.46447	0.53553

D

Accuracy: 0.5

Metric	AGAINST	FAVOR
Precision	0.333333	0.666667
Recall	0.5	0.5
F-measure	0.4	0.571429
Proportion in Training Data	0.466803	0.533197

E

Accuracy: 0.4537815096

Metric	AGAINST	FAVOR
Precision	0.434783	0.48
Recall	0.535714	0.380952
F-measure	0.48	0.424779
Proportion in Training Data	0.465358	0.534642

F

Accuracy: 0.552083333333

Metric	AGAINST	FAVOR
Precision	0.44	0.591549
Recall	0.275	0.75
F-measure	0.338462	0.661417
Proportion in Training Data	0.471316	0.528684

G

Accuracy: 0.482269504603

Metric	AGAINST	FAVOR
Precision	0.347368	0.76087
Recall	0.75	0.360825
F-measure	0.47482	0.48951
Proportion in Training Data	0.491706	0.508294

H

Accuracy: 0.379310339689

Metric	AGAINST	FAVOR
Precision	0.333333	0.428571
Recall	0.384615	0.375
F-measure	0.357143	0.4
Proportion in Training Data	0.466527	0.533473

I

Accuracy: 0.600000023842

Metric	AGAINST	FAVOR
Precision	0.625	0.5
Recall	0.833333	0.25
F-measure	0.714286	0.333333
Proportion in Training Data	0.464615	0.535385

J

Accuracy: 0.5365853644

Metric	AGAINST	FAVOR
Precision	0.619048	0.45
Recall	0.541667	0.529412
F-measure	0.577778	0.486486
Proportion in Training Data	0.45515	0.54485

K

Accuracy: 0.575471699238

Metric	AGAINST	FAVOR
Precision	0.509434	0.641509
Recall	0.586957	0.566667
F-measure	0.545455	0.60177
Proportion in Training Data	0.469852	0.530148

L

Accuracy: 0.518987346299

Metric	AGAINST	FAVOR
Precision	0.628571	0.431818
Recall	0.468085	0.59375
F-measure	0.536585	0.5
Proportion in Training Data	0.454746	0.545254

M

Accuracy: 0.58536585802

Metric	AGAINST	FAVOR
Precision	0.375	0.608108
Recall	0.09375	0.9
F-measure	0.15	0.725806
Proportion in Training Data	0.472868	0.527132

N

Accuracy: 0.605263161032

Metric	AGAINST	FAVOR
Precision	0.5	0.7
Recall	0.6	0.608696
F-measure	0.545455	0.651163
Proportion in Training Data	0.468849	0.531151

O

Accuracy: 0.714285731316

Metric	AGAINST	FAVOR
Precision	0.666667	0.733333
Recall	0.5	0.846154
F-measure	0.571429	0.785714
Proportion in Training Data	0.467842	0.532158

Bidirectional Encoding

Accuracy: 0.539134388317

A

Accuracy: 0.571428571429

Metric	AGAINST	FAVOR
Precision	0.622222	0.512821
Recall	0.595745	0.540541
F-measure	0.608696	0.526316
Proportion in Training Data	0.45727	0.54273

B

Accuracy: 0.346153846918

Metric	AGAINST	FAVOR
Precision	0.441176	0.272727
Recall	0.319149	0.387097
F-measure	0.37037	0.32
Proportion in Training Data	0.454245	0.545755

C

Accuracy: 0.428571432829

Metric	AGAINST	FAVOR
Precision	0.5	0.416667
Recall	0.125	0.833333
F-measure	0.2	0.555556
Proportion in Training Data	0.46447	0.53553

D

Accuracy: 0.666666686535

Metric	AGAINST	FAVOR
Precision	0	0.666667
Recall	0	1
F-measure	0	0.8
Proportion in Training Data	0.466803	0.533197

E

Accuracy: 0.521008400607

Metric	AGAINST	FAVOR
Precision	0.492958	0.5625
Recall	0.625	0.428571
F-measure	0.551181	0.486486
Proportion in Training Data	0.465358	0.534642

F

Accuracy: 0.583333333333

Metric	AGAINST	FAVOR
Precision	0	0.583333
Recall	0	1
F-measure	0	0.736842
Proportion in Training Data	0.471316	0.528684

G

Accuracy: 0.517730498568

Metric	AGAINST	FAVOR
Precision	0.328571	0.704225
Recall	0.522727	0.515464
F-measure	0.403509	0.595238
Proportion in Training Data	0.491706	0.508294

H

Accuracy: 0.379310339689

Metric	AGAINST	FAVOR
Precision	0.333333	0.428571
Recall	0.384615	0.375
F-measure	0.357143	0.4
Proportion in Training Data	0.466527	0.533473

I

Accuracy: 0.5

Metric	AGAINST	FAVOR
Precision	1	0.444444
Recall	0.166667	1
F-measure	0.285714	0.615385
Proportion in Training Data	0.464615	0.535385

J

Accuracy: 0.512195126313

Metric	AGAINST	FAVOR
Precision	0.583333	0.411765
Recall	0.583333	0.411765
F-measure	0.583333	0.411765
Proportion in Training Data	0.45515	0.54485

K

Accuracy: 0.641509436211

Metric	AGAINST	FAVOR
Precision	0.583333	0.689655
Recall	0.608696	0.666667
F-measure	0.595745	0.677966
Proportion in Training Data	0.469852	0.530148

L

Accuracy: 0.531645573393

Metric	AGAINST	FAVOR
Precision	0.625	0.435897
Recall	0.531915	0.53125
F-measure	0.574713	0.478873
Proportion in Training Data	0.454746	0.545254

M

Accuracy: 0.536585362946

Metric	AGAINST	FAVOR
Precision	0.428571	0.65
Recall	0.5625	0.52
F-measure	0.486486	0.577778
Proportion in Training Data	0.472868	0.527132

N

Accuracy: 0.684210529453

Metric	AGAINST	FAVOR
Precision	0.588235	0.761905
Recall	0.666667	0.695652
F-measure	0.625	0.727273
Proportion in Training Data	0.468849	0.531151

O

Accuracy: 0.666666686535

Metric	AGAINST	FAVOR
Precision	0.6	0.6875
Recall	0.375	0.846154
F-measure	0.461538	0.758621
Proportion in Training Data	0.467842	0.532158

Seen-Unseen Raw Results

Abortion as the Unseen Target

The size of testing data was 1698; 849 "AGAINST" posts and 849 "FAVOR" posts.

Linear SVM

- 811 classified as AGAINST
- 887 classified as FAVOR
- Accuracy: 0.52414605418138982
- Correctness (AGAINST): 426
- Correctness (FAVOR): 464

Conditional Encoding

- 995 classified as AGAINST
- 703 classified as FAVOR
- Accuracy: 0.50412249705535928
- Correctness (AGAINST): 501
- Correctness (FAVOR): 355

Bidirectional Encoding

- 810 classified as AGAINST
- 888 classified as FAVOR
- Accuracy: 0.51884570082449943
- Correctness (AGAINST): 421
- Correctness (FAVOR): 460

Gay Rights as the Unseen Target

The size of testing data was 998; 499 "AGAINST" posts and 499 "FAVOR" posts.

Linear SVM

- 479 classified as AGAINST
- 519 classified as FAVOR
- Accuracy: 0.51503006012024044

- Correctness (AGAINST): 247
- Correctness (FAVOR): 267

Conditional Encoding

- 577 classified as AGAINST
- 421 classified as FAVOR
- Accuracy: 0.48897795594168569
- Correctness (AGAINST): 283
- Correctness (FAVOR): 205

Bidirectional Encoding

- 579 classified as AGAINST
- 419 classified as FAVOR
- Accuracy: 0.50300601202404804
- Correctness (AGAINST): 291
- Correctness (FAVOR): 211

Marijuana as the Unseen Target

The size of testing data was 364; 182 "AGAINST" posts and 182 "FAVOR" posts.

Linear SVM

- 166 classified as AGAINST
- 198 classified as FAVOR
- Accuracy: 0.5439560439560439
- Correctness (AGAINST): 91
- Correctness (FAVOR): 107

Conditional Encoding

- 188 classified as AGAINST
- 176 classified as FAVOR
- Accuracy: 0.49450549450549453
- Correctness (AGAINST): 93
- Correctness (FAVOR): 87

Bidirectional Encoding

- 228 classified as AGAINST
- 136 classified as FAVOR
- Accuracy: 0.5439560439560439
- Correctness (AGAINST): 122
- Correctness (FAVOR): 76

Glossary

Although the first instance of each acronym is explained, this section has been provided so one can easily look up each acronym's definition.

LSA Latent Semantic Analysis. 12, 13

LSTM Long Short-Term Memory. i, v, 2, 3, 10–12, 17, 18, 26, 29, 32, 33, 35, 36

OCC one-class classifier. 35

OHV One-Hot Vector. 7, 8

SG Skip-Gram. v, 5, 7, 8

SVD singular-value decomposition. v, 12, 13

SVM Support Vector Machine. i, v, vii, 2, 5, 8–10, 17–26, 29, 32–36, 40, 45, 48, 54, 59

TF-IDF Term Frequency-Inverse Document Frequency. 4

WVM word vector model. 3, 5, 8