Louis Max Cley Slater

Sentiment Analysis of Texts on the Iraq War

Computer Science Tripos – Part II

Pembroke College

April 3, 2018

Proforma

Should be 1 page

Name:	
College:	**Pembroke College**
Project Title:	**Sentiment Analysis of Texts on the Iraq War**
Examination:	**Computer Science Tripos – Part II, July 2018**
Word Count:	**Max 12,000**
Project Originator:	Louis Max Cley Slater
Supervisor:	Dr Tamara Polajnar

Original Aims of the Project

Due to my interest in both natural language processing and politics and a lack of previous work done in the area, I decided that I wanted to perform sentiment analysis on British political texts. After extensive research, I found a study [18] that manually assessed the biases of British newspaper articles on the Iraq war, so decided that I would use the dataset produced by the study. I aimed to develop a program to retrieve the texts of the newspaper articles specified in the study [18] and implement a classifier using this data and a bag of words model.

Work Completed

- Problem with University's licence for DowJones (and others?). Didn't cover Scraping data/API use? Be specific and add Licence agreement(s) to bibliography - Switched to Hansard and voting datasets. Made the data retrieval stage lengthier. Cite datasets. - Numbers about success of classifier - Add anything else completed?

Special Difficulties

- Mention the original dataset licence issues and change?

Declaration

I, Louis Max Cley Slater of Pembroke College, being a candidate for Part II of the Computer Science Tripos, hereby declare that this dissertation and the work described in it are my own work, unaided except as may be specified below, and that the dissertation does not contain material that has already been used to any substantial extent for a comparable purpose.

Signed

Date

Contents

List of Figures

Acknowledgements

Chapter 1

Introduction

This dissertation describes the development of a system that uses various natural language processing techniques to analyse texts on the Iraq war. The texts are transcripts of relevant debates in the House of Commons between 11th September 2001 and 18th March 2003. Despite underlying difficulties of the task, the system produced gives consistently good results when evaluated using a variety of different techniques. The details of the implementation and evaluation are expanded upon throughout this document.

1.1 Motivation

The initial idea for this project was due to my interest in both natural language processing and politics. While computational approaches are often applied to political texts, very few studies have ever specifically concerned a British corpus (a collection of written or spoken material stored on a computer and used to find out how language is used [4]). After reading various papers that used natural language processing techniques on political corpora, I noted the most common studies concerned sentiment analysis of short texts, such as newspaper headlines or Tweets. This motivated me to investigate the task of performing sentiment analysis on longer pieces of text, to make the project more unique still.

On a personal note, the invasion of Iraq was the first political issue I engaged in; I attended anti-war protests with my mum when I was very young and have continued to closely follow the developments since then. It was an issue for which people had very strong views that were not determined by their political leanings [15], therefore making it a particularly interesting topic for sentiment analysis. Furthermore, after extensive research I couldn't find any research that carried out computational sentiment analysis which focused on war. The project I carried out was more relevant still due to the recent publication of the Chilcot Enquiry [3] and the ongoing situation in Iraq and Syria [19].

1.2 Corpora

The most essential component of any sentiment analysis system is its corpus. For this reason, the first task I carried out was to obtain the use of a relevant corpus that I was licensed to use.

1.2.1 Newspaper Dataset

The first potential dataset I found was that produced by Robinson, P. and Goddard, P. and Brown, R. and Taylor, P.M. in which they "evaluated media performance during the 2003 Iraq War" [18]. As part of their evaluation, they manually annotated the stance of 4,893 British newspaper articles on the Iraq war. They published the resulting dataset, but it didn't contain the body of the articles - only its headline, author, newspaper and publication date. I consequently investigated resources containing the text of the relevant articles and tried to cross-reference the data from these sources with the manually annotated stance. At the time, many newspapers published different stories online and in print, meaning that I could not rely on these. A few newspapers maintain electronic archives of their printed editions on the internet, however not enough newspapers had such archives. The final resource I looked into was Dow Jones Factiva, a "global news database" [12]. Upon inspection, this database contained the vast majority of the articles I needed and it was possible for me to cross-reference the articles in it with the labels annotated by Robinson, P. and Goddard, P. and Brown, R. and Taylor, P.M.. I initially accessed the dataset through the University of Cambridge's subscription. I therefore (falsely) assumed that this subscription would be sufficient for use in my project, however I later discovered that an academic licence did not permit me to use the API or to carry out text-mining. I consequently contacted Dow Jones and was told that the licence I required would cost in excess of $20,000.

1.2.2 Hansard

After exhausting all other options, I turned my attention to the House of Commons Hansard archives, which contains transcripts of debates between members of Parliament in the Commons Chamber [16]. Due to the licensing problems I encountered with Dow Jones Factiva [12], I immediately looked into the licence required to scrape data from the Hansard and found that it is covered by the Open Parliament Licence [8]. Since the Hansard archives are available under this licence, I was permitted to:

- "copy, publish, distribute and transmit the information"
- "adapt the information"
- "exploit the information commercially and non-commercially, for example, by combining it with other information, or by including it in your own product or application"

A further benefit of using the Hansard is the fact that it can be labelled using MPs' voting records - for example, if an MP voted in favour of the invasion of Iraq, all of their

speeches on Iraq can be labelled as pro-war. This allowed me to develop the sentiment analysis system using a supervised learning model.

1.3 Challenges

There are many flaws of our 'democracy' in the United Kingdom. In my opinion, one major flaw is our voting system; we have a first-past-the-post system in which each member of the electorate only gets one vote. This vote goes to a candidate who is (usually) a member of a political party [7]. This party will often force an MP to vote in accordance with the party line, regardless of the MP's own views. This subverts democracy when MPs vote with their party line, despite making contradictory election promises [5]. This issue was particularly prevalent leading up to the Iraq war, where despite protests [6] showing the public's overwhelming opposition to the invasion of Iraq, MPs voted in favour of the invasion. Within a month of being elected as Prime Minister in 1997, Tony Blair said "Mine is the first generation able to contemplate the possibility that we may live our entire lives without going to war or sending our children to war. That is a prize beyond value." [9], just six years before encouraging his Labour Party to vote invade Iraq. As a result, many Labour MPs who were previously opposed to the invasion voted in its favour. This hypocrisy of Members of Parliament is commonplace in British politics, meaning that as the electorate, we are frequently misled by the politicians representing us and the difficulties of party politics add to this.

At first glance, the Hansard appears to be a great resource for natural language processing, however its data is poorly presented and to use it, it's necessary to scrape the data from the inconsistent web pages, which in itself presented a difficulty. Having done so, labelling the data is not necessarily as straightforward as I suggested in 1.2.2, as MPs do not always vote consistently with their own views.

Manually labelling speeches is cumbersome and not possible within this project (give number of relevant speeches). Manually labelling MPs' stances is also very time consuming, as it would require a lot of research reading old newspaper articles, and even then, the stances of some of the lesser known MPs might still be ambiguous. Therefore, in this project I will have to use MPs' voting records to label their speeches. Due to reasons I have previously mentioned, MPs will not necessarily vote consistently with their own views, meaning that there will be some degree of noise in the data. This issue is particularly great for the Iraq war, since the debate split parties and the leadership of both the Labour Party and Conservative Party were in favour of the war; before the Iraq war vote on 18th March 2003, the leaders of the two largest parties, Tony Blair (Labour Party) and Iain Duncan-Smith (Conservative Party) used party whips to persuade MPs to vote to support the invasion.

One result of this project will be to provide evidence in support of one of the following hypotheses:

- *Hypothesis 1* : MPs who feel strongly enough about a given issue to speak about it in the House of Commons will not vote on their own convictions, rather than in accordance with their Party's line.

- *Hypothesis 2* : Political Parties can influence their own MPs to vote with the party line, regardless of the views of the individual MPs.

Admittedly, viewing the issues in such a binary way is an oversimplification and how an MP's party affects their vote will differently vary on a vote-by-vote basis for different MPs. While this simplification would be nave in a Politics dissertation, overlooking nuances will allow for greater technical discussion, relevant to Computer Science dissertation. In essence, the better the performance of the classifier produced, the greater weight will be given to Hypothesis 1 over Hypothesis 2. This noise inherent in the data presents difficulties which can be mitigated through the design of the classifier.

1.4 Previous Work

While many studies into sentiment analysis have been based around political issues, since 2009 the majority of such research has concerned Tweets. The first study to use Twitter as its primary corpus was 'Twitter power: Tweets as electronic word of mouth' [11] and there have since been countless studies following suit. In 2010, Pak, Alexander and Paroubek, Patrick proposed that Twitter could be used to determine public opinion [17], which was proven true later that year when sentiment analysis of Twitter provided predictions that paralleled the results of traditional election polls for the German federal election [20].

This focus on Twitter is useful, but since most political decisions are made in Government and not on the internet, I propose that we should scrutinise the opinions of our elected politicians more than Twitter users. Part of the reason that what our MPs do in Parliament is not considered as much as it should be is due to the inaccessibility of the House of Commons; the language used by MPs when debating is unnaturally formal, making it difficult to follow and unnecessarily long-winded, whereas Tweets are inherently short and easy to interpret. A sentiment analysis system essentially summarises a text, meaning that applying sentiment analysis to Parliamentary debates would be a useful stepping stone towards summarising a debate. Unfortunately, although it makes the project more interesting, analysing longer political texts presents more challenges than analysing Tweets, in part due to the lack of guidance from similar previous work.

In the U.S., there have been a small number of papers detailing sentiment analysis on transcripts of Congress debates [1] [10]. The results of these studies indicate that determining an MP's stance on the Iraq war from their speeches in the House of Commons may be possible, however these papers use transcripts to determine a politician's political party, which is likely to be more clear-cut than their stance on a particular issue.

The lack of relevant works to this project highlights its uniqueness, which is one of the principal motivations for the project.

Chapter 2

Preparation

There were three main stages to the preparation of the project:

1. Learning about the necessary concepts and methods. This was useful as it helped me to make informed decisions about implementation decisions. This required considerable work, as most of the skills and knowledge required to undertake the project are not taught in the Cambridge BA Computer Science course and the parts that are taught are Part II courses. The time scale of the Part II project meant that I had to learn the courses ahead of the lectures. Sections 2.1 through 2.7 detail this stage of the preparation and section 2.8 is also strongly linked to this stage.

2. Defining and planning the project. A project of this scale needs clear definition of its goals and a well defined plan designed to achieve these goals. Sections 2.9 through 2.11 detail this stage of the preparation.

3. Selecting the tools to be used for implementation. Section 2.12 details this stage of the preparation.

2.1 Introduction to Supervised Learning

A supervised learning problem the task of determining the label of a given input. This is split into two phases: Learning and predicting.

In the learning phase, the system receives inputs of feature vectors and their associated labels. A feature vector of length k is usually denoted by $\mathbf{x}$ where

$$\mathbf{x} = (x_1, x_2, ..., x_k) \quad \forall i \in \mathbb{Z}^+ . \forall x_i \in \mathbb{R}. \tag{2.1}$$

A feature vector contains encodes the information necessary to predict a label. In the context of this project, there is a feature vector for each speech we consider, which contains information about the words in the speech. The label is usually denoted by y. The set of values that y can take varies depending on the context of the supervised learning problem. For example, in a regression problem, $y \in \mathbb{R}$. This project concerns binary classification, since we simplify the problem so that we consider all speeches to be

either pro-war or anti-war. Because of this, from now on, we will only consider binary classification problems, that is where $y \in \{-1, +1\}$.

The supervised learning system creates a function h that takes a feature vector as an input and outputs a label. That is

$$h(\mathbf{x}) = y. \tag{2.2}$$

This definition allows us to intuitively view each feature vector as a point in k-dimensional space. We consider each point to be either negative ($y = -1$) or positive ($y = +1$). In this analogy, h is a function that determines whether a point is negative or positive, depending on where it is in the k-dimensional space. The more points that h sees, the better its estimation of whether new unseen points are negative or positive.

The figure below shows a visualisation of our intuition of feature vectors, where $k = 2$. In this diagram, the supervised learning system learns a function to distinguish the '-' and '+' points. Given a new, previously unseen point, this function would be able to estimate whether it is a '-' or a '+'.

2.2 Introduction to the Naïve Bayes Classifier

This is one of the simplest classifiers to understand and implement. It uses the assumption that all features are independent of each other:

$$p(x_i|x_j) = p(x_i) \quad \forall i, j \in \mathbb{Z}^+. \tag{2.3}$$

We say that the classifier is 'naïve' because of this assumption. Although the assumption is very rarely true, the classifier still provides good performance [14].

In addition to this assumption, the classifier uses Bayes Theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}. \tag{2.4}$$

The intuition behind the classifier is that given a set of features $\mathbf{x}$, we should assign it to the class that has the highest probability, given the set of features. Using the assumption of conditional independence and Bayes Theorem, we can compute this probability as

follows:

$$\begin{aligned}
p(C = y|\mathbf{x}) &= \frac{p(\mathbf{x}|C = y)p(C = y)}{p(\mathbf{x})} \\
&= \frac{p(x_1, \ldots, x_k|C = y)p(C = y)}{p(\mathbf{x})} \\
&= \frac{p(x_1, \ldots, x_k, C = y)}{p(\mathbf{x})} \\
&= \frac{p(x_1|x_2, \ldots, x_k, C = y)p(x_2, \ldots, x_k, C = y)}{p(\mathbf{x})} \\
&\vdots \\
&= \frac{p(x_1|x_2, \ldots, x_k, C = y)p(x_2|x_3 \ldots, x_k, C = y) \cdots p(x_k|C = y)}{p(\mathbf{x})} \\
&= \frac{p(x_1|C = y)p(x_2|C = y) \cdots p(x_k|C = y)}{p(\mathbf{x})} \\
&= \frac{\prod_{i=1}^{k} p(x_i|C = y)}{p(\mathbf{x})}.
\end{aligned} \tag{2.5}$$

Clearly, this shows that we can compute y using:

$$\begin{aligned}
y &= \underset{y}{\operatorname{argmax}} \left(\frac{\prod_{i=1}^{k} p(x_i|C = y)}{p(\mathbf{x})} \right) \\
&= \underset{y}{\operatorname{argmax}} \prod_{i=1}^{k} p(x_i|C = y).
\end{aligned} \tag{2.6}$$

We can estimate each $p(x_i|C = y)$ trivially using the training data. Given that in this project I am only considering binary classifiers, where $y \in -1, +1$, we can write this as:

$$max\left(\prod_{i=1}^{k} p(x_i|C = -1), \prod_{i=1}^{k} p(x_i|C = +1) \right). \tag{2.7}$$

Due to its simplicity and good performance, I will use the naïve Bayes classifier as a baseline for my project.

2.3 Introduction to Support Vector Machines

Support vector machines (SVMs) are widely used, state-of-the-art classifiers which were designed for binary classification (although they have since been modified to work for multi-class classification) [2]. Since I am viewing the task of determining the sentiment of speeches on the Iraq war as a binary classification problem, using a SVM is a natural choice.

In contrast to the naïve Bayes classifier, the SVM approach to classification is not inherently probabilistic. Instead, they are a form of maximum margin classifier. A

maximum margin classifier computes a hyperplane of the form

$$\mathbf{w} \cdot \mathbf{x} + b = 0. \tag{2.8}$$

where $\mathbf{w}$ is a normal to the hyperplane. $\mathbf{w}$ and b are determined by the maximisation (2.11) and $\mathbf{x}$ is a point on the hyperplane. This hyperplane separates the training data, so that for all positive examples

$$\mathbf{w} \cdot \mathbf{x} + b \geq 0 \tag{2.9}$$

and for all negative examples

$$\mathbf{w} \cdot \mathbf{x} + b < 0. \tag{2.10}$$

The idea of the maximum margin classifier is that it maximises δ, the distance between the hyperplane and the closest examples to it. That is, it computes

$$\underset{\mathbf{w},b}{\text{argmax}}(min(\delta)). \tag{2.11}$$

The figure below illustrates this problem in a 2D space (i.e. where $\mathbf{x} = (x_1, x_2)$)

To determine whether feature vector $\mathbf{x}_i$ should be positively or negatively labelled, we simply need to determine which side of the hyperplane it lies on. This gives us the decision function

$$y_i = \begin{cases} +1, & \mathbf{w} \cdot \mathbf{u} + b \geq 0 \\ -1, & \text{otherwise} \end{cases} \tag{2.12}$$

where $\mathbf{u}$ is the feature vector being classified. The support vectors are defined as the training examples that lie closest to the hyperplane. From the equation of the hyperplane (2.8), we see that we have the freedom to scale $\mathbf{w}$ and b by a constant factor without changing the hyperplane itself. We can therefore define this scaling by imposing the following constraint on all training examples for mathematical convenience:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \geq 0. \tag{2.13}$$

For the support vectors, we then have

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 = 0 \tag{2.14}$$

$$\implies \mathbf{w} \cdot \mathbf{x}_i = \frac{1}{y_i} - b. \tag{2.15}$$

$$\implies b = \frac{1}{y_i} - \mathbf{w} \cdot \mathbf{x}_i. \tag{2.16}$$

We now need to compute the width of the margin so we can then form an expression to maximise it. The figure above gives us some intuition as to how we can achieve this. Since $\mathbf{w}$ is perpendicular to the hyperplane, $\frac{\mathbf{w}}{||\mathbf{w}||}$ must be the unit normal to the

hyperplane. We can then use a positively labelled support vector, $\mathbf{x}_+$ and a negatively labelled support vector, $\mathbf{x}_-$ to get an expression for the margin width:

$$\begin{aligned} \text{Margin width} &= \frac{\mathbf{w}}{||\mathbf{w}||} \cdot (\mathbf{x}_+ - \mathbf{x}_-) \\ &= \frac{\mathbf{w} \cdot \mathbf{x}_+ - \mathbf{w} \cdot \mathbf{x}_-}{||\mathbf{w}||} \end{aligned} \tag{2.17}$$

We can now substitute in the result from (2.15) to give

$$\begin{aligned} \text{Margin width} &= \frac{\left(\frac{1}{y_+} - b\right) - \left(\frac{1}{y_-} - b\right)}{||\mathbf{w}||} \\ &= \frac{\frac{1}{y_+} - \frac{1}{y_-}}{||\mathbf{w}||} \\ &= \frac{1+1}{||\mathbf{w}||} \\ &= \frac{2}{||\mathbf{w}||} \end{aligned} \tag{2.18}$$

Our goal is to maximise the width given by (2.18). For mathematical convenience, we can instead solve the equivalent problem of minimising $\frac{1}{2}||\mathbf{w}||^2$. This optimisation is subject to the constraints in (2.14). In order to solve this constrained optimisation problem, we must use Lagrange multipliers

$$\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n) \tag{2.19}$$

where n is the number of training examples. This results in the Lagrange function

$$L = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{n} \alpha_i (y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1). \tag{2.20}$$

Our task is now to solve the unconstrained maximisation problem

$$(\mathbf{w}_{opt}, b_{opt}) = \underset{\mathbf{w},b}{\operatorname{argmax}}(L) \tag{2.21}$$

Using the Karush-Kuhn-Tucker conditions, we can show that $\alpha_i = 0$ for all feature vectors that are not support vectors [2]. This results in fast computation and means that after training, we only need to store the support vectors. Therefore, from now on, we will sum over $\mathcal{S}$, the set of indices corresponding to the support vectors.

In order to solve the optimisation problem defined in (2.21), we must find the partial derivative of L with respect to both $\mathbf{w}$ and b, setting the resulting expressions to 0 (since we want to vary $\mathbf{w}$ and b in order to find the maximum L).

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i \in \mathcal{S}} \alpha_i y_i \mathbf{x}_i = 0 \tag{2.22}$$

$$\implies \mathbf{w} = \sum_{i \in \mathcal{S}} \alpha_i y_i \mathbf{x}_i \tag{2.23}$$

$$\frac{\partial L}{\partial b} = \sum_{i \in \mathcal{S}} \alpha_i y_i = 0 \tag{2.24}$$

We can now substitute (2.23) into (2.20) to obtain a new expression for L (and simplify using (2.24)) as follows:

$$\begin{aligned} L &= \frac{1}{2}(\sum_{i\in\mathcal{S}} \alpha_i y_i \mathbf{x}_i)(\sum_{j\in\mathcal{S}} \alpha_j y_j \mathbf{x}_j) - \sum_{i\in\mathcal{S}} \alpha_i y_i (\sum_{j\in\mathcal{S}} \alpha_j y_j \mathbf{x}_j) \cdot \mathbf{x}_i - b\sum_{i\in\mathcal{S}} \alpha_i y_i + \sum_{i\in\mathcal{S}} \alpha_i \\ &= \frac{1}{2}(\sum_{i\in\mathcal{S}} \alpha_i y_i \mathbf{x}_i)(\sum_{j\in\mathcal{S}} \alpha_j y_j \mathbf{x}_j) - (\sum_{i\in\mathcal{S}} \alpha_i y_i \mathbf{x}_i)(\sum_{j\in\mathcal{S}} \alpha_j y_j \mathbf{x}_j) + \sum_{i\in\mathcal{S}} \alpha_i \\ &= \sum_{i\in\mathcal{S}} \alpha_i - \frac{1}{2}(\sum_{i\in\mathcal{S}} \alpha_i y_i \mathbf{x}_i)(\sum_{j\in\mathcal{S}} \alpha_j y_j \mathbf{x}_j) \\ &= \sum_{i\in\mathcal{S}} \alpha_i - \frac{1}{2}\sum_{i\in\mathcal{S}}\sum_{j\in\mathcal{S}} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j). \end{aligned} \tag{2.25}$$

We now need to find the values $\boldsymbol{\alpha}$ which maximise L:

$$\boldsymbol{\alpha}_{opt} = \underset{\boldsymbol{\alpha}}{\operatorname{argmax}}(L). \tag{2.26}$$

I won't go into the details of how to find these values, but this can be done numerically. Further to this, it can be shown that the space of L is convex, so we will not find a local maximum. This is a significant advantage of using SVMs over neural networks.

We can then find $\mathbf{w}_{opt}$ by substituting $\boldsymbol{\alpha}_{opt}$ into (2.23) and using the support vectors and their labels. From this, we can find b_{opt} using (2.16) and substituting in $\mathbf{w}_{opt}$ along with any support vector and its label. We can substitute our values for $\boldsymbol{\alpha}_{opt}$ and b_{opt} into the initial decision rule to obtain a new decision rule:

$$y_i = \begin{cases} +1, & \sum_{i\in\mathcal{S}} y_i (\boldsymbol{\alpha}_{opt})_i (\mathbf{x}_i \cdot \mathbf{u}) + b \geq 0 \\ -1, & \text{otherwise.} \end{cases} \tag{2.27}$$

Thus far, we have been working under the assumption that our data is linearly separable. In practice, this is very rarely the case and for this project due to the inherent noise in our data (described in §1.3), this assumption is very unlikely to hold.The figure below illustrates a simple example for which the data are not linearly separable.

In order to fix this problem, we can use a transformation, ϕ, to transform our feature vectors into a new space in which our data is more easily separable. Applying this transformation to (2.25) gives us a new L:

$$L = \sum_{i\in\mathcal{S}} \alpha_i - \frac{1}{2}\sum_{i\in\mathcal{S}}\sum_{j\in\mathcal{S}} \alpha_i \alpha_j y_i y_j (\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)). \tag{2.28}$$

We maximise this as before, finding a new $\boldsymbol{\alpha}_{opt}$ from (2.26) and then using those values to find b_{opt}. We can then use these values of $\boldsymbol{\alpha}_{opt}$ and b_{opt} along with the transformation ϕ to obtain another decision rule:

$$y_i = \begin{cases} +1, & \sum_{i\in\mathcal{S}} y_i (\boldsymbol{\alpha}_{opt})_i (\phi(\mathbf{x}_i) \cdot \phi(\mathbf{u})) + b \geq 0 \\ -1, & \text{otherwise.} \end{cases} \tag{2.29}$$

We are yet to define ϕ, but if we consider the contexts in which it is used, we see that it is always in the form $\phi(\mathbf{x}) \cdot \phi(\mathbf{x}')$. Therefore, rather than define ϕ itself, we define a kernel function

$$k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x}) \cdot \phi(\mathbf{x}'). \tag{2.30}$$

From this, we can rewrite (2.28) and (2.29) as:

$$L = \sum_{i \in \mathcal{S}} \alpha_i - \frac{1}{2} \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{S}} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \tag{2.31}$$

$$y_i = \begin{cases} +1, & \sum_{i \in \mathcal{S}} y_i (\boldsymbol{\alpha}_{opt})_i k(\mathbf{x}_i, \mathbf{u}) + b \geq 0 \\ -1, & \text{otherwise.} \end{cases} \tag{2.32}$$

The choice of kernel function can have a significant effect on the performance of a SVM. In order to ensure good results, I will train the SVM using different kernel functions, so the classifier learns which kernel function will work best for the data. The three kernel functions I will consider are:

$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x} \cdot \mathbf{x}' \tag{2.33}$$

$$k(\mathbf{x}, \mathbf{x}') = (\gamma(\mathbf{x} \cdot \mathbf{x}') + r)^d \qquad \gamma \in \mathbb{R}_{>0}. r \in \mathbb{R}_{\geq 0}. d \in \mathbb{N}_{>0} \tag{2.34}$$

$$k(\mathbf{x}, \mathbf{x}') = e^{-\gamma ||\mathbf{x} - \mathbf{x}'||^2} \qquad \gamma \in \mathbb{R}_{>0} \tag{2.35}$$

From now on, I will refer to (2.33), (2.34) and (2.35) as the linear kernel, the polynomial kernel and the radial basis function (rbf) kernel respectively. Using the linear kernel is equivalent to our SVM before we introduced the transformation ϕ. This shows that even with kernel functions, our data may not be linearly separable. It can be shown that linear and polynomial kernels do not necessarily transform the data so that into a space for which it is linearly separable, but the rbf kernel can always map the feature vectors to a space where they are linearly separable. This means that for the linear and polynomial kernels, we may not be able to produce a classifier with the given constraints and for the rbf kernel the SVM is very susceptible to overfitting. Both of these problems can be solved by introducing soft-margins to our SVM. This means that we will allow the SVM to incorrectly classify some of the training examples. In order to do this, we introduce a parameter C which trades off correct classification of training examples with a greater margin width. A greater margin width results in a smoother function, so means that the SVM is less likely to overfit. The higher the value of C, the more training examples the SVM will fit correctly. C is a hyperparameter - that is a parameter whose value is fixed before the classifier is trained. All of the hyperparameters for each of the kernels we are considering are shown in the table below. The hyperparameter choice significantly effects the performance of the SVM, so we need an algorithm for choosing them. This is discussed further in both §2.4.2 & §3.3.

Kernel	Hyperparameters
Linear	C
Polynomial	C, γ, r, d
Radial basis function	C, γ,

2.4 Introduction to Evaluating Supervised Learning Systems

2.4.1 Train/Test Split

In §2.1, we saw how a supervised learning system is trained on one set of data (the training set), then this trained model is used to predict the labels of previously unseen data (the testing set). In order to evaluate a system, we require the actual data labels, so we can assess the accuracy of the predictions. Having training examples in the testing set results will not provide a useful measure of the system's performance, as it would unfairly reward overfitting. In order to overcome this, we must split our labelled data before we start developing a model. Using 90% for training and 10% for testing is the most common way to split the labelled data.

2.4.2 Cross Validation

It is useful to split the training set into k disjoint folds. Doing this means that we can iterate the process of training and evaluating without using the data set aside for testing. This is done by training on $k-1$ of the folds, then evaluating the system on the fold left out. This is repeated k times, so each fold is used for evaluation exactly once. Averaging over all the folds each fold gives a reliable evaluation metric. We can use this method to determine the system's hyperparameters and any other settings that need to be determined before training. This is called cross-validation. To do this, we repeat the method described for different combinations of hyperparameters and settings and select the combination whose average evaluation metric is greatest.

2.4.3 Evaluation Metrics

Thus far, we have only spoken abstractly about an evaluation metric. There are various options and choice of the measure should be context-specific. The basis of the definitions used in most metrics are defined in the table below:

		Prediction	
		Positive	**Negative**
Actual Value	**Positive**	True Positive	False Negative
	Negative	False Positive	True Negative

From this, we can now define the following quantities:

$$TP = \text{Number of True Positives} \tag{2.36}$$
$$FN = \text{Number of False Negatives} \tag{2.37}$$
$$FP = \text{Number of False Positives} \tag{2.38}$$
$$TN = \text{Number of True Negatives} \tag{2.39}$$

Accuracy is the simplest evaluation metric. We define this as:

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN} \tag{2.40}$$

If we have an unbalanced dataset (which is the case in this project), then accuracy is not a good evaluation metric. To illustrate this, consider the case where 1% of our data is positive and 99% of our data is negative. If we had a classifier that always predicted that an example was negative, its accuracy would be 99%. Since accuracy is not a useful measure for unbalanced datasets, such as the dataset I am using in this project, I will not consider it any further.

We now define two further measures:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.41}$$
$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.42}$$

A good evaluation metric for an unbalanced dataset will incorporate some trade-off between precision and recall. Such a metric may give greater weighting to precision for a precision-critical task or greater weighting to recall for a recall-critical task. Since this project is neither precision-critical nor recall-critical, we can use the F_1 score as the evaluation metric, since the F_1 score is defined as the harmonic mean of precision and recall:

$$F_1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{2.43}$$

2.5 Introduction to the Bag-of-Words Model

In mathematics, a bag is a synonym for a multiset - an abstract data type which is like a set, but differs in that it can contain duplicates. In this project, I will represent MPs' speeches using a bag-of-words model, meaning that the representation ignores the order of words. Despite its simplicity, the bag-of-words model is used successfully in a range of sentiment classification applications, as it can effectively capture the discourse of a text [13].

To illustrate a bag-of-words model using an example, I will use an quote from a House of Commons debate on 18th March 2003:

"The best way to avoid war is to work through the United Nations."
- Bill Tynan (Labour Party)

In a simple bag-of-words implementation, where case and punctuation are ignored, this sentence would be stored as:

{the : 2,
to : 2,
best : 1,
way : 1,
avoid : 1,
war : 1,
is : 1,
work : 1,
through : 1,
united : 1,
nations : 1}.

It is important to note that the order of the elements above is not relevant.

2.6 Introduction to the House of Commons

2.7 Spam Email Dataset - DEADLINE : 2ND APRIL

2.8 Requirements Analysis

In my initial Project Proposal [Appendix A] I outlined the project as

2.9 Implementation Approach

2.10 Software Engineering Techniques

2.11 Choice of Tools

2.12 Starting Point

2.13 Summary - DEADLINE: 4TH APRIL

Chapter 3

Implementation

3.1 Data Scraper

3.2 Database - DEADLINE: 6TH APRIL

3.3 Classifier

3.4 Summary - DEADLINE: 11TH APRIL

Chapter 4

Evaluation Code - DEADLINE: 16TH APRIL

Chapter 5

Evaluation

5.1 Unit Testing

5.2 Internal Evaluation - DEADLINE: 18TH APRIL

5.3 External Evaluation

5.4 Evaluation of Project Goals

5.5 Summary - DEADLINE: 20TH APRIL

Chapter 6

Conclusions

6.1 Achievements

6.2 Lessons Learned

6.3 Future Work - DEADLINE: 23RD APRIL

Chapter 7

Diagrams - DEADLINE: 25TH APRIL

Bibliography

[1] Maneesh Bhand, Dan Robinson, and Conal Sathi. Text classifiers for political ideologies. 2009.

[2] C. M Bishop. *Pattern recognition and machine learning.* Information science and statistics. Springer, New York, 2006.

[3] John Chilcot et al. The report of the iraq inquiry. *Report of a Committee of Privy Counsellors. London, UK: House of Commons*, 2016.

[4] Cambridge Dictionary. Corpus dictionary definition.

[5] A. Feldman, B. Harris, and J. Urban. Government tracker.

[6] Carmen Fishwick. 'we were ignored': anti-war protesters remember the iraq war marches.

[7] UK Government. Current state of the parties.

[8] UK Government. Open parliament licence v3.0.

[9] The Guardian. Tony blair's key quotes.

[10] Mohit Iyyer, Peter Enns, Jordan Boyd-Graber, and Philip Resnik. Political ideology detection using recursive neural networks. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1113–1122, 2014.

[11] Bernard J. Jansen, Mimi Zhang, Kate Sobel, and Abdur Chowdury. Twitter power: Tweets as electronic word of mouth. *Journal of the American Society for Information Science and Technology*, 60(11):2169–2188.

[12] Dow Jones. Factiva.

[13] Dan Jurafsky and James H. Martin. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition / Daniel Jurafsky and James H. Martin.* Prentice Hall series in artificial intelligence. 2nd ed., international ed. edition, 2009.

[14] Kevin P. Murphy. *Machine learning [electronic resource] : a probabilistic perspective / Kevin P. Murphy.* Adaptive computation and machine learning series. MIT Press, Cambridge, MA, 2012.

[15] BBC News. Did your mp support the rebels?

[16] House of Commons. Hansard archives.

[17] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, volume 10, 2010.

[18] P. Robinson, P. Goddard, R. Brown, and P.M. Taylor. Content and framing study of united kingdom media coverage of the iraq war, 2003.

[19] Kareem Shaheen. Us-led coalition says its strikes have killed 800 iraqi and syrian civilians.

[20] Andranik Tumasjan, Timm Oliver Sprenger, Philipp G Sandner, and Isabell M Welpe. Predicting elections with twitter: What 140 characters reveal about political sentiment. *Icwsm*, 10(1):178–185, 2010.

Appendix A

Project Proposal

Assessors like to see some sample code or example circuit diagrams, and appendices are the sensible places to include such items. Accordingly, software and hardware projects should incorporate appropriate appendices. Note that the 12,000 word limit does not include material in the appendices, but only in extremely unusual circumstances may appendices exceed 10-15 pages - if you feel that such unusual circumstances might apply to you you should ask your Director of Studies and Supervisor to apply to the Chairman of Examiners. It is quite in order to have no appendices. Appendices should appear between the bibliography and the project proposal.

Computer Science Tripos – Part II – Project Proposal

Sentiment Analysis of British Newspaper Articles on the Iraq War

Louis Slater, Pembroke College

Originator: Louis Slater

12th October 2017

Project Supervisor: Dr Tamara Polajnar
Director of Studies: Dr Anil Madhavapeddy
Project Overseers: Dr Timothy Griffin & Professor Anuj Dawar

Introduction

While there have been lots of studies involving sentiment analysis of political texts to determine their bias, none of these have uniquely involved British newspaper articles. Furthermore, after extensive research, I have not found any sentiment analysis of articles to determine their stance on a war. The purpose of this project is to develop a program that can reasonably determine the stance of any British newspaper article on the Iraq war. The core part of this project will be developing a program that achieves this using a bag-of-words method.

Starting point

In the past few years, there has been a lot of research into determining political biases of shorter segments of text, such as Tweets in the 2010 paper by Pak and Paroubek on Twitter as a Corpus for Sentiment Analysis and Opinion Mining. On the other hand, Political Ideology Detection Using Recursive Neural Networks by Iyyer, Enns, Boyd-Graber and Resnik uses a corpus containing longer texts US Congressional floor debate transcripts. Although there are clear differences between these transcripts and the newspaper articles that I plan to use (for example, the fact that the transcripts were initially spoken, whereas the articles were not), there are also many similarities (for example the length and inherently political nature of the corpora). Since this study showed that a bag-of-words method can successfully determine the bias of these transcripts with a 65% accuracy, it is justified to use a similar model to determine the bias of the newspaper articles I shall analyse.

The corpus I will use will be articles on the Iraq war from up to seven of the UKs most popular national daily newspapers and their Sunday equivalents published between 16th March 2003 and 18th April 2003. I will use a database of these articles compiled by Robinson, Goddard, Brown and Taylor in their 2003 study, Content and Framing Study of United Kingdom Media Coverage of the Iraq War, in which they manually determine the stance of 4,893 news articles from seven British daily newspapers and their Sunday equivalents (Daily Telegraph, The Times, The Guardian/The Observer, The Independent, The Daily Mail, The Mirror, The Sun/News of the World). This database does not include the articles texts, so the first part of my implementation will be to scrape this data from as many of these articles as possible. I will be able to get these texts from existing online Newspaper archives. I have already found searchable archives for The Guardian, The Observer, The Daily Telegraph, The Sunday Telegraph, The Independent, Indy on Sunday, The Times and the Sunday Times, all of which I will be able to use. Scraping textual data from the other newspapers in the database may be prove more difficult, but I will as many possibilities as I feasibly can within the scope of the project.

Resources required

In addition to the database and archives mentioned above, I will also require the use of a computer. I intend to mainly use my own computer, which has an Intel Core i7 processor and runs Windows 10. I will use the computing facilities in my college if my laptop is lost, broken or stolen. I will back up my work using both Google Drive and GitHub, which I will also use as a version control repository. I may also require the use of a server or external hard drive to store the corpus I use; however, this will be dependent on the amount of data that I scrape in the initial part of my project.

Work to be done

The project breaks down into the following sub-projects:

1. Gaining access to as many of the relevant searchable newspaper archives as possible.
2. Scraping data from as many articles as possible in the database compiled by Robinson, Goddard, Brown and Taylor.
3. Implementing a program to determine the biases of texts on the Iraq war, using the corpus I gather, along with corresponding the Reporters Tones from the database compiled by Robinson, Goddard, Brown and Taylor.
4. Running the program on the texts and comparing the results with the manually determined biases to judge the effectiveness of the program.

Success citeria

The project will be a success if I develop a program that can determine the stance of an article on the Iraq war with a greater than 50% accuracy.

Possible extensions

If I meet my success criteria early, I shall attempt one, or both, of the following extensions:

- Implementing a program that performs the same function as the initial program I develop, but using a different method, such as a recursive neural network. If I complete this extension, I will be able to compare the effectiveness of the two methods.
- Extrapolating the results using new datasets and analysing these results. Possible datasets I could use are newspaper articles from different countries, publications or times or transcripts of parliamentary debates.

Timetable

Planned starting date is the beginning of Michaelmas Week 3 (Thursday 19th October 2017).

1. **Michaelmas week 3** Gain access to as many of the relevant searchable newspaper archives as possible.
2. **Michaelmas weeks 4–5** Scrape data from as many articles as possible in the database compiled by Robinson, Goddard, Brown and Taylor, creating a database of the texts, their manually determined bias and other relevant information on them. If necessary, I will also get access to a server and store the database I compile on this server.

3. **Michaelmas weeks 6–8** Implement a program to determine the biases of texts on the Iraq war, using the corpus I gather, along with corresponding the Reporters Tones from the database compiled by Robinson, Goddard, Brown and Taylor.

4. **Michaelmas vacation** Finish the implementation, then run the program on the texts and compare the results with the manually determined biases to judge the effectiveness of the program.

5. **Lent weeks 1–2** Write the progress report and start work on possible extensions of the project.

6. **Lent weeks 3–4** Finish the extensions to the project.

7. **Lent weeks 5–6** Write the first draft of the dissertation.

8. **Lent weeks 7–8** Revise the dissertation in accordance with feedback I receive from my supervisor.

9. **Easter vacation** Finish revising the dissertation and submit the final project.