What's in a Word? Detecting Partisan Affiliation from Word Use in Congressional Speeches

Ulya Bayram, John Pestian, Daniel Santel and Ali A. Minai, *Senior Member, IEEE*

***Abstract*—Politics is an area of broad interest to policy-makers, researchers, and the general public. The recent explosion in the availability of electronic data and advances in data analysis methods – including techniques from machine learning – have led to many studies attempting to extract political insight from this data. Speeches in the U.S. Congress represent an exceptionally rich dataset for this purpose, and these have been analyzed by many researchers using statistical and machine learning methods. In this paper, we analyze House of Representatives floor speeches from the 1981 - 2016 period, with the goal of inferring the partisan affiliation of the speakers from their use of words. Previous studies with sophisticated machine learning models has suggested that this task can be accomplished with an accuracy in the 55 to 80% range, depending on the year. In this paper, we show that, in fact, very comparable results can be obtained using a much simpler linear classifier in word space, indicating that the use of words in partisan ways is not particularly complicated. Our results also confirm that, over the period of study, it has become steadily easier to infer partisan affiliation from political speeches in the United States. Finally, we make some observations about specific terms that Republicans and Democrats have favored over the years in service of partisan expression.**

I. Introduction

Detecting political ideology or partisan affiliation from political speeches is interesting in several ways. The degree to which an analysis based on patterns of word usage can accurately detect the partisan affiliation of the speakers is an indication of whether partisanship is clearly delineated in the language they use. This, in turn, says something about specific politicians and about the general degree of partisanship in the politics of the time. In its details, such analysis can also help identify specific words or word combinations that have become associated with particular ideologies, or are being pushed by certain politicians. This can be useful, both to political campaigns and to voters trying to understand what they are being told implicitly.

Political speeches occur in many contexts: In the legislature, during campaigns, and at ceremonial events. Of these, the first category is very well-documented because most legislatures, including the United States Congress, maintain official records of everything that is said on the floor. These records are a treasure trove for political scientists and historians, and have been analyzed extensively. More recently, methods from natural language processing (NLP) and machine learning (ML) have also been applied to this task [1]–[6]. Among the analyzed political records in the literature are those from the British Parliament [7] and the Irish Dail debates [8], but the most commonly studied data are of U.S. House of Representatives and Senate speeches [1]–[3], [6], [8]–[10]. The speeches have been analyzed from many perspectives, including partisanship [1], [3], [7], agreement [8], ideology [6], etc. The present study considers the task of identifying party affiliation from speeches made in the U.S. House of Representatives, with a particular emphasis on word usage. House speeches are believed to be more partisan than Senate speeches [1], which motivates the choice adopted in this study.

Ulya Bayram (bayramua@mail.uc.edu) is with the Department of Electrical Engineering and Computer Science, University of Cincinnati, Cincinnati OH 45221 and Cincinnati Children's Hospital Medical Center, Cincinnati, OH 45229. John Pestian is with University of Cincinnati Department of Pediatrics and Cincinnati Children's Hospital Medical Center. Daniel Santel is with Cincinnati Children's Hospital Medical Center, Cincinnati, OH 45229. Ali A. Minai (Ali.Minai@uc.edu) is with the Department of Electrical Engineering and Computer Science, University of Cincinnati, Cincinnati OH 45221.

Acknowledgement: The authors thank Biomedical Informatics, Department of Pediatrics in Children's Hospital Medical Center.

II. Background and Motivation

The primary motivation for the work presented in this paper is to address three questions:

1) To what degree is it possible to infer party affiliation from analyzing the texts of floor speeches, and how difficult is this problem?
2) Can this inference be based on the use of specific words, or are higher order features such as word combinations helpful?
3) What does the possibility of inferring party affiliation from speeches of successive Congresses say about the dynamics of partisanship in American politics?

To address these questions, data from several Congresses was analyzed using a suite of binary classification models, ranging from very simple to quite complex. The purpose was not necessarily to find an optimal classifier, but to probe the nature of the task by applying the same set of tools to several datasets ranging over a long period of time. The data for this analysis came from a study by Getzkow, Shapiro and Taddy [9].

Most previous studies on the classification of political speeches have used a bag-of-words approach, taking unigrams (single words) to be the representational features [1], [2], [7], [8], [10], though some have utilized bi-grams or tri-grams in the hope of capturing politically relevant "phrases" [3], [9]. Most previous studies [1]–[3], [6], [8]–[10] have assumed – justifiably – that inferring partisan classification from speech texts is a difficult task. This has generally been borne out by the results of those studies, which have typically achieved classification accuracies in the range of 55 to 75

percent, depending on the year and the way the question is framed. In general, however, these results were achieved using quite sophisticated machine learning algorithms – in particular, support vector machines with added feature pre-processing [1], [2], [10]. In this study, we use SVM as one of the classifiers, but also include several others that are generally simpler. The goal is to determine whether an SVM or a classifier of similar complexity is really necessary, or if the same results can be achieved with simpler classifiers.

The following sections describe the methods used, present comparative results from all classifiers on several datasets, and look at the variation in partisanship over time.

III. Methods

A. Data and Pre-Processing

The collection of speeches made available in the study by Getzkow, Shapiro and Taddy [9] are used in this study[1]. This set contains material from bound editions for the 43rd to the 111th Congresses, and daily editions for the 97th to the 114th (where the 114th lacks a couple of months' data). In this study, the daily edition is used, ranging from the 97th to the 114th Congresses, restricted to only seven selected Congresses (97th, 100th, 103rd, 106th, 109th, 112nd, 114th) and only to House speeches. Speeches given by each speaker each day are extracted and saved separately. Speeches of less than 1KB size are removed, and 6,000 speeches from each set are randomly selected. These 6,000 speeches (3,000 Democrats, 3,000 Republicans) selected from each Congress are then randomly separated into 4,000 training and 2,000 test sets (half Democratic and half Republican in each set), thus giving balanced training and testing sets from each Congress.

To keep the study simple, we have opted to use unigrams as the primary features in the comparison of models. This is consistent with the choice made in most previous studies. However, all classifiers used take joint usage of words into account in simple or complex ways. Our classifiers also include feed-forward neural networks with both small and large hidden layers that are, in principle, capable of inferring more complex features through learning, as well as the use of a lexical network approach called *excess weight density* (EWD) classification, that considers word combinations explicitly. A more fundamental reason for using unigrams as the primary features is that it provides insight into the degree to which particular words are used to partisan ends by politicians. This is the simplest level at which the use of partisan language can be explored, and also the easiest to understand.

The documents are pre-processed by removing all punctuation and by converting to lower-case. Stemming or stopword removal are *not* applied because of the concern that they can reduce discrimination. However, addressing phrases ('Mr./Madam Speaker' or 'Mr./Madam Chairman/Chair') are removed. Also, after unigram extraction from the training sets, those words with less than 50 occurrences and those occurring in fewer than ten training set speeches are removed from each set. The vocabulary sizes (the number of unigrams) after these pre-processing steps for each dataset are shown in Table I. The data pre-processing for the construction of the lexical network in the EWD classifiers is different. Here, all punctuation *except end of sentence marks* is removed, and end-of-sentence marks are replaced by the same sign. Also, in order to obtain meaningful lexical networks, English language stopwords provided by the NLTK package [11] are removed.

TABLE I: The vocabulary sizes (the number of unigrams) obtained from the training set of each Congress.

Congress	97th	100th	103rd	106th	109th	112th	114th
Size	2,211	2,286	2,174	2,156	2,297	2,151	2,015

Every speech is represented by a feature vector, where each feature corresponds to a distinct word. For all classifiers other than the two EWD classifiers, the feature vector is the vector of the log of word counts in the speech normalized to be of unit length. For the EWD classifiers, each speech is coded as a binary vector, i.e., the ith feature has value 1 if the ith word occurs in the speech, and 0 if it does not.

B. Classifiers

This study uses nine classifiers, as described below. Of these, seven are trained using standard supervised learning methods, while two are variations on the EWD approach developed in a previous study that showed good performance on classifying suicidal mindsets [12]. Supervised learning is done using the "Adam" optimizer [13], which is used in various benchmark experiments and recommended as a possible default [14]–[16]. A learning rate of 0.001 and batch size of 50 is used in all cases. The classifiers are all trained for 50 epochs, which was empirically determined to be sufficient and avoided overfitting.

The following classifiers are used in this study:

- **Random Forest (RF):** As an ensemble classifier, this method generates a selected number of decision trees on various sub-samples of the given data [17]. In this study, the RF implementation available in the scikit-learn package [18] is used with the number of trees set to 500. Other parameters are kept at default, and used to classify the unigram features.
- **Support Vector Machine (SVM):** Support vector machines [19] are among the most widely used classifiers for NLP applications. In this study, the implementation in the scikit-learn package [18] is used. As correct parameter selection is very important for the efficient use of this classifier, a grid-search is done to select the kernel function, cost parameter, and gamma (a coefficient used by the kernel functions), and the best set of parameters which returned the highest accuracy on 8-fold classification of training data is used on tests.
- **Least Mean-Square (LMS):** This is one of the oldest and simplest classifiers available [20], using a single neuron. For an input feature vector $x = [x_1\ x_2\ ...\ x_n]$, the input to the unit is given by $y(x) = \sum_{j=1}^{n} w_j x_j + \theta$, where w_j

[1] https://data.stanford.edu/congress_text

is the weight of the jth feature and θ is a trainable bias. The output is given by $z(x) = tanh(y)$. The artificial neural network (NN) library of tensorflow [21] is used for training this with gradient descent, using mean-squared error as the loss function. During training, the target output for speeches by Democrats is 1 and for those by Republicans -1. After training, the inferred class is determined to be "Democrat" if $z \geq 0$ and "Republican" otherwise. Thus, LMS is a linear classifier.

- **Feed-Forward Neural Network with 20 Hidden Neurons and No Dropout (NN20-ND):** This classifier is a single hidden layer feed-forward neural network with 20 hidden neurons, where each layer is completely connected to the layer before it. The hidden neurons also use the hyperbolic tangent activation function. There are two output neurons, and outputs are calculated via the softmax function. The artificial neural network (NN) library of tensorflow [21] is used for training the network with gradient descent, using cross-entropy as the loss function. No regularization is used in this system.
- **Feed-Forward Neural Network with 20 Hidden Neurons and High Dropout (NN20-D):** This is the same as the previous classifier, but it is trained using a very high dropout regularization rate of 0.98, with the intention of forcing hidden layer neurons to learn distinct features. The training process and training parameters are the same as for NN20-ND.
- **Feed-Forward Neural Network with 1000 Hidden Neurons and No Dropout (NN1000-ND):** This classifier is like NN20-ND, but has 1000 hidden neurons. The purpose is to see if allowing a large number of intermediate features could potentially improve classification – even at the risk of some overfitting. There is no dropout or other regularization.
- **Feed-Forward Neural Network with 1000 Hidden Neurons and High Dropout (NN1000-D):** This is the same as the previous classifier, but uses a high dropout rate of 0.98. The purpose is to see whether distinct complex features would emerge if a large number of hidden neurons are made available but forced to be distinct via dropout.
- **Excess Weight Density with No Spreading (EWD-NS):** Explained in more detail in a previous study [12], this approach builds lexical networks from the training sets and checks which lexical network a given document is more consistent with. Specifically, counting the co-occurrences of the words i and j in the same sentence, it computes a correlation-coefficient for each word pair:

$$a_{ij}^q = \frac{p_{ij}^q - p_i^q p_j^q}{\sqrt{p_i^q(1-p_i^q)}\sqrt{p_j^q(1-p_j^q)}} \tag{1}$$

where p_{iq} is the fraction of sentences in which the words co-occur, and p_i is the fraction of sentences that includes word i. The lexical network is an undirected network where each node is a word from the dataset vocabulary and the edge weight between nodes i and j is given by a_{ij} if that is positive and 0 otherwise. A separate lexical network is constructed for each class using the training set. In this case, we get networks G_D and G_R for the "Democrat" and "Republican" classes, respectively. The normalized mean weight density of each network is given by:

$$d(G_q) = \frac{\sum_{i,j \in G_q} w_{i,j}}{n(n-1)/2} \tag{2}$$

where n is the number of nodes (i.e., the size of the dataset vocabulary) and G_q could be G_D or G_R.
Given a test document, V, all the words in it are detected and the nodes corresponding to these are activated, thus defining activated sub-networks, $E_D(V)$ and $E_R(V)$, respectively, in both lexical networks. The normalized mean weight density of each active sub-network is computed using the following equation:

$$d(E_q(V)) = \frac{\sum_{i,j \in E_q(V)} w_{i,j}}{m(m-1)/2} \tag{3}$$

where $m \leq |V|$ is the number of nodes (i.e., the number of unique words in the document matching the nodes) and E_q could be E_D or E_R.
The *excess weight density* (EWD) for the document in each network is calculated as

$$\delta_q(V) = d(E_q(V)) - d(G_q). \tag{4}$$

This method of measuring connection strength of a network is commonly called "density" or "weight density" as it is evaluated by the weight values [22], [23]. The EWD quantifies the degree to which the strength of connectivity within each activated sub-network for the document exceeds the value that would be expected for a random sub-network of the same size extracted from the full lexical network. Classification is performed by using:

$$\Delta(V) = \delta_D(V) - \delta_R(V). \tag{5}$$

If $\Delta(V)$ is non-negative, the document is classified as D, else as R.

- **Excess Weight Density with Spreading Activation (EWD-S):** An extension of the EWD-NS method, this method additionally uses spreading activation as studied in various domains [24]–[27]. Also explained in more detail in the previous study [12], this method activates the matching test words on the trained lexical networks as in EWD, but also lets the activation spread to neighboring nodes receiving sufficient stimulus from activated ones. Thus, the activated sub-networks obtained in this approach are larger than those obtained in the EWD-NS approach. The goal behind this approach is to capture neighboring ideas and thoughts present in the lexical networks, which might enhance the classification of the test data. The amount of spread is controlled by a parameter.

In this study, a small amount of spread is allowed in order to include only the most relevant neighbors.

IV. Results and Discussion

TABLE II: 97th Congress (1981-1982).

Methods	Precision	Recall	Accuracy
RF	0.602	0.619	0.605
SVM	0.620	0.639	**0.623**
LMS	0.620	0.596	0.615
NN20-ND	0.568	0.606	0.573
NN20-D	**0.623**	0.607	0.619
NN1000-ND	0.562	0.600	0.566
NN1000-D	0.605	**0.641**	0.611
EWD-NS	0.603	0.511	0.587
EWD-S	0.601	0.528	0.588

TABLE III: 100th Congress (1987-1988).

Methods	Precision	Recall	Accuracy
RF	0.622	0.613	0.620
SVM	0.634	0.658	0.639
LMS	**0.652**	0.600	0.640
NN20-ND	0.604	0.561	0.597
NN20-D	0.639	**0.677**	**0.647**
NN1000-ND	0.581	0.613	0.585
NN1000-D	0.639	0.621	0.635
EWD-NS	0.647	0.517	0.617
EWD-S	0.641	0.520	0.614

TABLE IV: 103rd Congress (1993-1994).

Methods	Precision	Recall	Accuracy
RF	0.673	0.671	0.672
SVM	**0.707**	**0.722**	**0.711**
LMS	0.697	0.710	0.701
NN20-ND	0.662	0.699	0.671
NN20-D	0.693	0.717	0.700
NN1000-ND	0.672	0.663	0.669
NN1000-D	0.699	0.696	0.698
EWD-NS	0.701	0.501	0.643
EWD-S	0.699	0.488	0.639

TABLE V: 106th Congress (1999-2000).

Methods	Precision	Recall	Accuracy
RF	0.651	**0.699**	0.662
SVM	0.682	0.696	**0.685**
LMS	0.678	0.667	0.675
NN20-ND	0.634	0.657	0.638
NN20-D	0.678	0.678	0.678
NN1000-ND	0.634	0.642	0.635
NN1000-D	0.675	0.672	0.674
EWD-NS	**0.700**	0.575	0.664
EWD-S	0.692	0.588	0.663

A. Results for Individual Datasets

Tables II through VIII show the results of applying all classifiers to each of the seven datasets, and Table IX gives the cumulative performance of each classifier over all datasets. Several interesting observations can be made from these:

1) Overall, SVM has the best performance, though for every dataset, there is at least one other model that comes close in performance.

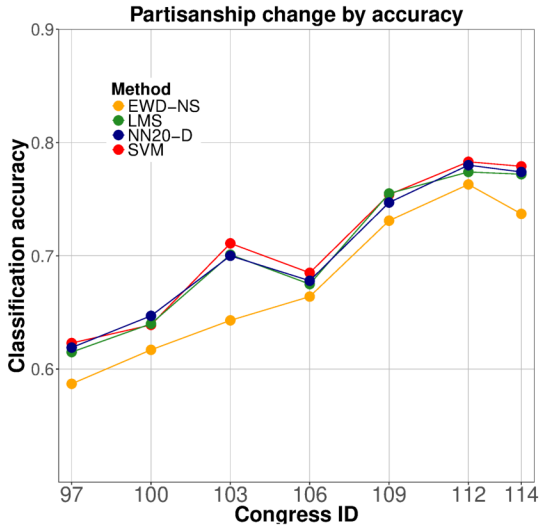
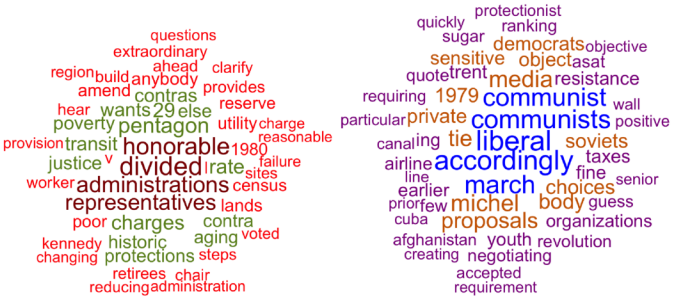


Fig. 1: Accuracy of SVM, LMS, NN20-D, and EWD-NS for all the selected Congresses. The change is an implicit indicator of partisanship in speeches, with increased classification accuracy suggested more partisan use of words.



(a) Democrat (b) Republican

Fig. 2: 97th Congress House top words according to the LMS classifier.



(a) Democrat (b) Republican

Fig. 3: 100th Congress House top words according to the LMS classifier.

TABLE VI: 109th Congress (2005-2006).

Methods	Precision	Recall	Accuracy
RF	0.759	0.701	0.739
SVM	0.768	0.730	**0.754**
LMS	0.758	**0.751**	**0.755**
NN20-ND	0.703	0.708	0.704
NN20-D	**0.774**	0.699	0.747
NN1000-ND	0.686	0.697	0.689
NN1000-D	0.742	0.749	0.744
EWD-NS	0.736	0.720	0.731
EWD-S	0.724	0.712	0.720

TABLE VII: 112th Congress (2011-2012).

Methods	Precision	Recall	Accuracy
RF	0.763	0.751	0.759
SVM	0.785	**0.779**	**0.783**
LMS	0.778	0.767	0.774
NN20-ND	0.742	0.735	0.739
NN20-D	**0.803**	0.742	0.780
NN1000-ND	0.728	0.738	0.731
NN1000-D	0.777	0.777	0.777
EWD-NS	0.789	0.719	0.763
EWD-S	0.781	0.714	0.757

TABLE VIII: 114th Congress (2015-2016).

Methods	Precision	Recall	Accuracy
RF	0.751	0.705	0.735
SVM	0.780	**0.778**	**0.779**
LMS	0.773	0.771	0.772
NN20-ND	0.736	0.743	0.738
NN20-D	**0.805**	0.724	0.774
NN1000-ND	0.720	0.734	0.724
NN1000-D	0.769	0.766	0.768
EWD-NS	0.763	0.688	0.737
EWD-S	0.738	0.689	0.722

TABLE IX: Cumulative results across all selected Congresses.

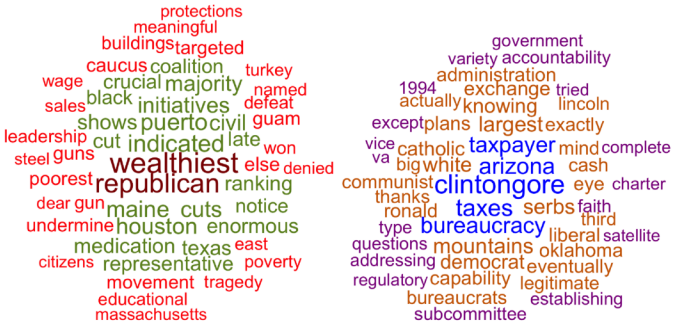
Methods	Precision	Recall	Accuracy
RF	0.686	0.681	0.685
SVM	0.709	**0.715**	**0.711**
LMS	0.709	0.695	0.705
NN20-ND	0.664	0.673	0.666
NN20-D	**0.713**	0.692	0.707
NN1000-ND	0.654	0.670	0.657
NN1000-D	0.700	0.703	0.701
EWD-NS	0.708	0.604	0.678
EWD-S	0.700	0.604	0.673

2) The performance of a 20 hidden unit neural network with very high dropout is very close to that of SVM. This is not unexpected. However, it is interesting that dropout has a consistently beneficial effect in this classifier, indicating that the diversity induced in the hidden neurons by dropout captures some useful information.
3) The neural network with a very large hidden layer has worse performance than the network with a smaller hidden layer, though dropout is again quite beneficial.
4) Remarkably, the performance of the simplest algorithm – LMS, which uses only a single tanh neuron – is consistently close to SVM, indicating that almost all the predictability achieved by SVM is essentially a linear classification in the space of unigrams (single words). Any advantage provided by learning more complex features is small. However, a closer look reveals another notable fact: While LMS achieves high precision, its recall is consistently significantly lower than SVM, and accounts for all of its small disadvantage against the latter. Since the Democratic class is coded as "positive" in this study, lower recall indicates that LMS consistently classifies more Democratic speeches as Republican. In a more detailed study, it will be interesting to see if these false negatives have some specific attributes, e.g., if they



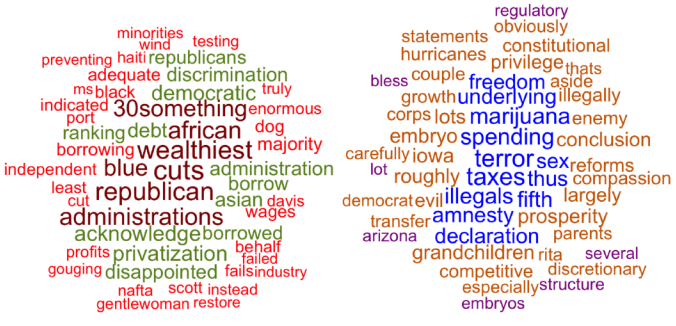
(a) Democrat (b) Republican

Fig. 4: 103rd Congress House top words according to the LMS classifier.



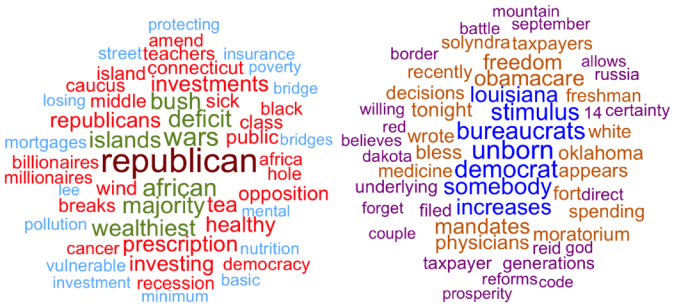
(a) Democrat (b) Republican

Fig. 5: 106th Congress House top words according to the LMS classifier.



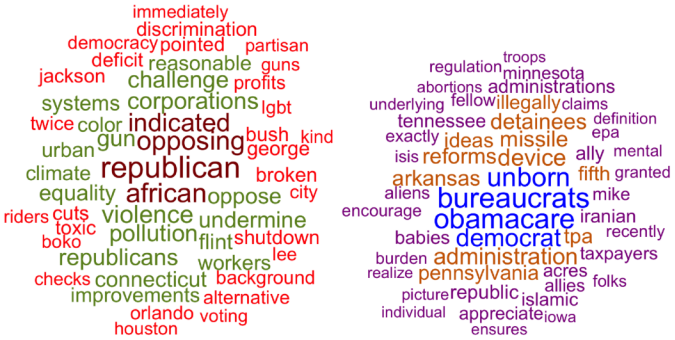
(a) Democrat (b) Republican

Fig. 6: 109th Congress House top words according to the LMS classifier.



(a) Democrat (b) Republican

Fig. 7: 112th Congress House top words according to the LMS classifier.



(a) Democrat (b) Republican

Fig. 8: 114th Congress House top words according to the LMS classifier.

represent speeches by more conservative Democrats.

5) RF has consistently poorer performance than SVM, LMS and NN20-D (see more on this below.)
6) The EWD algorithms both have performance significantly below SVM, LMS and the NN methods, though comparable with RF. Spreading activation actually makes things slightly worse. However, it is worth noting that the EWD approach does not use any optimization, and simply calculates second-order word statistics. As such, it is a very naive algorithm, and the fact that it can achieve performances fairly close to the best supervised learning methods again shows that most of the discriminative information in these datasets is easily accessible, albeit only enough to reach classification levels between 60-78 percent.
7) Overall, results in Table IX show that SVM, LMS, NN20-D and NN1000-D all have cumulative performance slightly above 70 percent accuracy, with SVM doing the best by a small amount.

B. Comparative Analysis of Classification Patterns

Table X shows in more detail where pairs of classifiers agreed or disagreed in their classifications on data from the 97th Congress. It shows that SVM, LMS, and NN20-D make very similar assignments (high values at diagonals), while EWD-NS appears to provide a different classification from the rest, with least agreement with the ML classifiers. However, the presence of significant off-diagonal components suggests that combining several classifiers, including EWD-NS, could further improve performance.

TABLE X: Pairwise comparison of classification between classifiers for the 97th Congress test data. Each 2×2 box represents two classifiers: r = row label, c = column label. The cells within each box show the following: cell (1,1) = number of test points classified correctly by both r and c; cell (2,2) = number misclassified by both r and c; cell (1,2) = number classified correctly by r and misclassified by c; cell (2,1) = number misclassified by r and classified correctly by c. Colors indicate the correct class of the data points: Blue = Democrats; Red = Republicans.

	SVM		LMS		NN20-D		EWD-NS	
SVM	639 608		554	85	550	89	398	241
		361 392	42	319	57	304	113	248
LMS			596 635		562	45	391	205
	551 57	84 308		404 365	34	359	120	284
NN20-D					607 632		401	206
	547 61	85 307	600 32	35 333		393 368	110	283
EWD-NS							511 663	
	462 146	201 191	500 135	163 202	506 126	157 211		489 337

C. Temporal Analysis

Figure 1 shows the performance of four algorithms – SVM, LMS, NN20-D, and EWD-S – over all the datasets. The plot shows a clear upward trend in performance, indicating that partisan affiliations of Representatives have become more obvious in the language of their speeches over the period 1981 through 2015. This corroborates the intuitive perception of most voters, and extends a phenomenon noted in the previous studies to more recent Congresses [9]. The fact that all classifiers except the EWD ones track each other over time suggests that the change mainly involves the use of different words.

A notable deviation in the upward trend occurs in the 103th Congress, which covers the years 1993-94. This shows a "bubble" in the amount of partisanship – presumably related to the election of Bill Clinton as President and the Republican campaign on Contract with America that led to a huge wave election in 1994. It is also interesting to note that, while all unigram-based classifiers see the bubble identically, the EWD classifiers, which consider word combinations, do not. This suggests that the change was reflected more in the polarized use of specific words, but any definitive conclusion will require more detailed analysis.

D. Analysis of Salient Features

The fact that the very simple LMS model performs so well presents an interesting opportunity to evaluate the salience of

specific words underlying the classification. Since LMS uses a neuron with a monotonically increasing activation function (tanh), this can be done simply by looking at the magnitude and polarity of weights from individual words after learning. Figures 2 through 8 show wordclouds of the top positive (D) and negative (R) features found by LMS for each dataset. Several notable facts stand out:

1) Use of the word "Republican" has a high value in identifying speeches by Democrats from the 106th Congress onwards, while the word "Democrat" does not have similar usage among Republicans until later, and even then not to the same degree. The word "liberal" does appear prominently in Republican speeches during the Reagan years.
2) The salient features on the Republican side seem to be somewhat more policy-oriented, with terms such as "taxes", "bureaucracy", "spending", "marijuana", "terror", "unborn", "Obamacare", etc., whereas the Democratic speakers, in addition to the epithet "Republican", give more importance to "wealthiest" and "African-American", reflecting two important themes in the Democratic Party.
3) The word "communist" seems to have become important in Republican speeches in the late 1980s through the early 1990s, after which it fades away, presumably reflecting the end of the Cold War.

E. Comparison with Other Studies

It is difficult to make a direct comparison between this study and previous ones, since they all use different datasets and set up the classification task in somewhat different ways. However, an approximate comparison can be made with the studies that use one or more of the specific datasets used by us, and by looking at the ultimate classification accuracy.

Table XI shows results from four studies on classification of the U.S. House speech data. The study of Thomas, Pang, and Lee [10] looks at House speeches from the 109th Congress (which we also consider). However, it selects only the debates conducted on the most "controversial bills" in year 2005 whereas the data we use includes a large selection of speeches from 2005 and 2006. They group speeches by the debate topic, and attempt to classify if the speaker is likely to support/oppose a given bill given their previous voting behavior. Using unigram features, a graph-based previous agreement framework, and an SVM classifier, they reach a maximum accuracy of 80% on their selected test set, while the accuracy on the rest of their tests vary between 66%–76%. Another study [1] uses speeches from the 101st to 109th Congresses of both the Senate and House, where speeches of the 25 most liberal and 25 most conservative senators are selected, and party memberships are used as the ground truth. They experiment with unigram and tf-idf features in combination with SVM and naive Bayes classifiers, obtaining accuracies in the range 70%–80% using leave-one-out classification of selected 2005 House speeches. They also train their classifiers on 2005 House speeches and test them on Senate speeches from 1989 to 2006. They get 50-60% for 1989, then a drop to 40-50% in 1994, followed by 50-70% until 2002, a slight drop in 2002, and then an increase to the 60-80% range. These results agree qualitatively with ours, though these authors asked a very different question. A few of the most important unigrams found in this study in House speeches of year 2005 according to their SVM classifier (using either unigrams or tf-idfs) match with top words found by LMS on the 109th Congress, which includes 2005 and 2006 speeches (Figure 6). Among these few matching top Republican class words are "taxes", "freedom", "embryo", and "terror", while matching Democratic words are "republican", "cuts", and "administration".

A similar study was conducted by Jensen et al. [3], using House speeches from the 43rd to 110th Congresses, spanning the years 1873 to 2009. They extracted tri-gram *phrases* and evaluated which of them best distinguish party memberships. Using a simple statistical method to evaluate party membership by the selected phrases' usage frequencies, they obtain out-of-sample performance for each year, and their classifications range greatly between 40% to above 80% for a few tests, while the majority of the classifications appear to be between 55%–75%.

The study of Iyyer et al. [6] follows a different approach, and utilizes recurrent neural networks (RNN) with pre-trained word2vec word embeddings [28], [29]. Unlike previous studies, they attempt to solve a more difficult task: Detecting bias from sentences, and obtain ground truth labels for the sentences of this dataset by crowdsourcing. Using the same 2005 House speech data, they return 70% classification accuracy, which is lower than our findings in Table VI, showing the difficulty of sentence-level classification.

TABLE XI: Some studies using U.S. House speeches, and performances they reached on detecting party affiliations.

Study	Data	Performance
Thomas, Pang, and Lee [10]	House109 (2005)	80% accuracy
Yu, Kaufmann, and Diermeier [1]	House109 (2005)	80% loo accuracy
Iyyer, Enns, Graber, and Resnik [6]	House109 (2005)	70% accuracy
Jensen, Kaplan, Naidu, and Wilse-Samson [3]	House43rd–110th	$\sim$40%-$\sim>$80%

V. Conclusion

This paper has analyzed seven corpora of U.S. House of Representatives speeches over a 23 year period. The results point to several interesting conclusions.

Based both on the results of our study and comparison with previous studies, it seems that a very simple linear model (LMS) can achieve a classification performance very close to other sophisticated methods – at least on questions related to partisan affiliation. This suggests that looking for complex, multi-term features does not add much in this task. The consensus of all the studies, including ours, is that the accuracy of partisan classification from congressional speeches is bound by a roughly 55-80% accuracy. This is reasonable, since word-based classifiers look at only the most superficial level of

meaning. It will be interesting to extend these studies to include more sophisticated methods such as semantic embedding and recurrent neural networks, though early results even on these methods have not shown any improvement [6]. We also see a steady increase with time in the ability of the same classifiers to detect partisan affiliation, with a notable bubble around 1993-94. In general, this trend suggests that the use of words by American politicians has steadily become more partisan, with members of each party increasingly preferring their own words and memes. This also is in broad agreement with the results of others [9].

The high performance of the linear LMS classifier is achieved in spite of the fact that it is notably deficient in recall, though excellent on precision. This offers an interesting opportunity to explore possible asymmetries between word usage by politicians of the two parties. In particular, it may be a way to identify the rhetorical positioning of individual politicians, and to see if their votes correlate with their rhetoric.

This study deliberately confined itself mostly to standard classifiers and simple features. Future work will extend this in several ways. First, we intend to use the data from the remaining Congresses, and Senate data in addition to House data, to build a more complete picture of the situation. Second, we plan to apply classifiers that look at deeper semantic structure, sentiments, and cognitive factors to push classification further. The trial with EWD was a step in this direction. If more powerful methods still do not lead to better classification, it may indicate that some aspects of political debate are too complex to be picked up by machine learning. We have not done a systematic analysis of where the various classifiers tried in this study make errors, but it is possible that performance can be increased by combining classifiers. This is a direction we will explore in future studies. Finally, we intend to extend this work to other interesting questions, such as detecting state of mind (e.g., suicidal vs. non-suicidal [12]), ideological leanings, media bias, etc.

REFERENCES

[1] B. Yu, S. Kaufmann, and D. Diermeier, "Classifying party affiliation from political speech," *Journal of Information Technology & Politics*, vol. 5, no. 1, pp. 33–48, 2008.

[2] D. Diermeier, J.-F. Godbout, B. Yu, and S. Kaufmann, "Language and ideology in congress," *British Journal of Political Science*, vol. 42, no. 1, pp. 31–55, 2012.

[3] J. Jensen, S. Naidu, E. Kaplan, L. Wilse-Samson, D. Gergen, M. Zuckerman, and A. Spirling, "Political polarization and the dynamics of political language: Evidence from 130 years of partisan speech [with comments and discussion]," *Brookings Papers on Economic Activity*, pp. 1–81, 2012.

[4] A. Qaroush, I. M. Khater, and M. Washaha, "Identifying spam e-mail based-on statistical header features and sender behavior," in *Proceedings of the CUBE International Information Technology Conference*. ACM, 2012, pp. 771–778.

[5] Y. Sim, B. D. Acree, J. H. Gross, and N. A. Smith, "Measuring ideological proportions in political speeches," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 91–101.

[6] M. Iyyer, P. Enns, J. Boyd-Graber, and P. 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)*, vol. 1, 2014, pp. 1113–1122.

[7] A. Peterson and A. Spirling, "Classification accuracy as a substantive quantity of interest: Measuring polarization in westminster systems," *Political Analysis*, vol. 26, no. 1, pp. 120–128, 2018.

[8] B. E. Lauderdale and A. Herzog, "Measuring political positions from legislative speech," *Political Analysis*, vol. 24, no. 3, pp. 374–394, 2016.

[9] M. Gentzkow, J. M. Shapiro, and M. Taddy, "Measuring polarization in high-dimensional data: Method and application to congressional speech," National Bureau of Economic Research, Tech. Rep., 2016.

[10] M. Thomas, B. Pang, and L. Lee, "Get out the vote: Determining support or opposition from congressional floor-debate transcripts," in *Proceedings of the 2006 conference on empirical methods in natural language processing*. Association for Computational Linguistics, 2006, pp. 327–335.

[11] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*. O'Reilly Media, 2009.

[12] U. Bayram, A. A. Minai, and J. Pestian, "A lexical network approach for identifying suicidal ideation in clinical interview transcripts," in *International Conference on Complex Systems*. Springer, 2018, pp. 165–172.

[13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[14] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International conference on machine learning*, 2015, pp. 2048–2057.

[15] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "Draw: A recurrent neural network for image generation," *arXiv preprint arXiv:1502.04623*, 2015.

[16] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[17] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

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

[19] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[20] B. Widrow and M. Lehr, "30 years of adaptive neural networks: perceptron, madaline, and backpropagation," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1415–1442, 1990.

[21] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[22] K. A. Frank, "Identifying cohesive subgroups," *Social Networks*, vol. 17, no. 1, pp. 27–56, 1995.

[23] J. Moody and D. R. White, "Structural cohesion and embeddedness: A hierarchical concept of social groups," *American Sociological Review*, vol. 68, pp. 103–127, 2003.

[24] J. R. Anderson, "A spreading activation theory of memory," *Journal of Verbal Learning and Verbal Behavior*, vol. 22, no. 3, pp. 261–295, 1983.

[25] ——, "Act: A simple theory of complex cognition," *American Psychologist*, vol. 51, no. 4, p. 355, 1996.

[26] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin, "An integrated theory of the mind," *Psychological Review*, vol. 111, no. 4, p. 1036, 2004.

[27] P. Pirolli, *Information foraging theory: Adaptive interaction with information*. Oxford University Press, 2007.

[28] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *ICLR Workshop*, 2013.

[29] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS'2013*, 2013.