Developing an Automated Pipeline for Information Extraction and Speech Classification from Floor Debate Records of the House of Representatives of the Philippines

Carlos Alberto L. Arcenas
Ateneo de Manila University
Quezon City
carlos.arcenas@obf.ateneo.edu

Miguel N. Galace
Ateneo de Manila University
Quezon City
miguel.galace@obf.ateneo.edu

Marlene M. De Leon, Ph.D.
Ateneo de Manila University
Quezon City
mmana@ateneo.edu

ABSTRACT

A lot can be learned about members of Congress in the way they speak and vote on bills during plenary proceedings. This paper tackles the problem of automating the extraction of speech and vote data from legislative documents of the House of Representatives of the Philippines to create a classification model for determining votes based on given speeches. To accomplish this, a framework was conceptualized and developed to extract relevant data from structured documents. The framework was then applied to the official House website to gather documents which, in turn, coursed through automated text processing and data structuring. The resulting structured data is made accessible through an API, named the Floorreader API, to make the extracted information adaptable for further exploration. The resulting structured data was used to train a congressional vote classification model. Overall, 1,161 Congressional Records and 728 House Journals were sourced for processing, resulting in the collection of 112,736 speeches across the debate of 752 bills, paired with 1,649 votes. The classification model developed was able to achieve performance measures in precision, recall, and F1-score of up to 85%, 84%, and 84% respectively.

CCS CONCEPTS

• **Computing methodologies** → **Information extraction**; • **Information systems** → *Extraction, transformation and loading*; *Data cleaning*; • **Applied computing** → *Document management and text processing*; **E-government**; Event-driven architectures;

KEYWORDS

Information extraction, data collection, text tagging, open data, government analysis

1 INTRODUCTION

1.1 Context of the Study

In the year 2014, in accordance with Joint Memorandum Circular No. 2014-01 regarding the need to make government information more accessible to the general public, the Philippine Government launched the Open Data Philippines initiative with the goal of heralding "transparent, accountable, and participatory governance through open government data" [1]. The site, accessible at data.gov.ph, hosts a wealth of information, including summaries, statistics, and visualizations on offices ranging from the Bangko Sentral ng Pilipinas to the Commission on Filipinos Overseas. Currently, there is no data on the Congress of the Philippines available on the site, but different legislative documents are available to the public through the official Senate and House of Representatives websites.

At the same time, there exists a lack of sufficient voter education in the Philippines, with constituents basing their votes for officials across all levels of government on superficial elements such as a candidate's personality and prestige over more substantive factors such as platform and policy [15]. The general public is doing little to educate itself, and the lack of more digestible chunks of information on the sectors of government is doing little to alleviate the situation. As such, in light of the upcoming midterm elections for the entire House of Representatives and half of the Senate, the need for a targeted voter education resource based on the words and legislative actions of Congress members is evident, especially towards building a more informed electorate.

While there are many sources of information on the proceedings of Congress, such as the reports created by mainstream news outlets, there is no source of verbatim speeches (or transcripts) covering whole sessions made available in a usable and machine-readable format. Instead, any sources currently available only include certain points of interest, such as debates on controversial bills. In fact, the only source with this kind of comprehensive information are the legislative documents hosted on the official website of the House of Representatives (congress.gov.ph). However, this information is not made available in an easy-to-digest format suitable for textual analysis: a lot of work needs to be done to transform these long documents originally made for print and physical reading to a format easily used for other purposes.

1.2 Significance of the Study

As of writing, the Congress of the Philippines does not make such information on plenary proceedings available in machine-readable formats, such as JSON objects, XML, or even plain text. This is in stark contrast to the accessibility afforded in more technologically-developed countries. For instance, certain Commonwealth countries, such as the United Kingdom, Canada, and Australia, have online versions of parliamentary proceedings (called Hansard) available in Extensible Markup Language (XML) form; similarly, the United States has verbatim transcripts of debates in their houses of Congress available through government-maintained application programming interfaces (APIs) and other sources maintained by non-governmental organizations. To capacitate fruitful analysis of Philippine political discourse—especially on the national level—a framework (and an efficient implementation of this framework) is needed to extract and process data currently only available in human-readable forms.

Philippine Computing Journal Dedicated Issue on Natural Language Processing, pages 9–20
Vol. XIII No. 1 August 2018

This study will also follow in the footsteps of other systems aiming to organize and make sense of political information in the country. In particular, this study will build upon a system earlier made for documents concerning a smaller portion of government, the Senate Blue Ribbon Committee [6]. This study will follow the top-down approach (i.e., sourcing data directly from government sources) in order to make what would otherwise be obscure and inaccessible data relevant and meaningful to the average citizen.

One immediate application for such data can be the development of a speech classification model that will be trained with a vote-tagged speech corpus in order to predict the future votes of representatives on bills based on their speeches before they are even cast. The model can also be used to tag the likely votes of representatives on bills where no voting data is provided by the records. This will allow the general public a more intimate understanding of the positions of representatives across a breadth of issues. This is valuable information that would otherwise not be available without the development of such a classification model.

Beyond that, the API provided by this study for retrieving speech and vote data from plenary sessions in the House aims to spawn a host of applications that will use this data either for further research in the sentiment analysis field or simply to create tools that will aid voter education on the House of Representatives.

2 REVIEW OF RELATED LITERATURE

2.1 Information Extraction

With its wide purview, information extraction (IE) covers unstructured data documents of all types, with studies making use of IE techniques to source unique entities from letters, newspaper articles, police reports, and more. However, accurate information extraction relies on data being sourced from documents relatively free from error in terms of writing, and those that adhere to standards on abbreviation and other concerns [8]. Given that, many IE studies source data from official and maintained sources, such as police reports for crime information extraction [11], and political speeches for stance extraction and analysis.

In nations where speech transcripts are readily available, information retrieval becomes a trivial step in a wider study often concerning textual analysis, and is often a precursor to more involved methods of textual preprocessing, such as stemming and stop-word removal [9] in light of a wider analysis. In places where this is not the case, bespoke systems need to be created to extract textual data from documents to act as a data source before extraction can begin.

Information extraction and the wider field of knowledge management allows for other applications to emerge, such as the creation of sentiment analysis systems. Beyond more simplistic political discussions accessible online, certain governments make publicly available transcripts of their legislative floor debates. One particular study took interest in the formal debate found in digitized and catalogued records from the Congress of the United States of the America. Using this data, they created models for analysis that went beyond techniques involving simple document-level sentiment-polarity classification by taking into account how individual speeches are part of a wider debate or dialogue between two or more parties [17]. This study enabled a wide gamut of further research after they made their annotated data publicly available online, furthering the goal of achieving improved and more accurate information extraction methods.

One local study has already taken advantage of the growing availability of digitized government data in the Philippines. In a construction of an e-Legislation system for the Senate Blue Ribbon Committee, two approaches to data sourcing identified in an earlier study [13] were implemented—the "ground-up" approach, which gathers comments made by constituents on the efficacy of the Committee in regards to the measures it attempts to put into action, and the "top-down" approach, which involves the organization of government office-generated information for public dissemination—in order to reconcile and connect actions and opinion towards better governance [6]. Documents sourced by the latter approach were stored in a bespoke document management system capable of storing multiple types of resources, such as invites, minutes of meetings, memos of Senate activity, and more. These documents, once needed for further analysis, were then queried from a relational database and preprocessed to extract only the relevant textual information for use in modelling and sentiment analysis. The resulting cleaned data from the documents was then stored in another database for archiving and quick retrieval [6].

Another local study sought to extract key information from criminal case documents available on lawphil.net, which hosts a composite of semi-structured texts of different styles written by varying authors. The study adopted and modified, for their purposes, a framework for information extraction called "The Generic Information Extraction System." The modules comprising their method included: a text zoner, a filter, named entity recognition, POS tagging, and more. To test their system, the researchers had 50 documents manually annotated and compared these results with their automated method in determining relevant fields such as case number, laws cited, and court decision. They found their system to be fairly accurate, with measurements involving precision, recall, and F-measure all above 90%[7].

From these studies and many others, a common thread between IE studies can be found through their data processing and storage mechanisms: while the specific implementation of each system differs based on the context, the underlying principles remain the same. Each system can be divided into two sections: the processing and identification section, where text is extracted and entities are named and identified, and the storage section, where the source data is put into a document management system, and the extracted data is put into another storage solution, often a relational database designed to address specific queries on the data extracted [3]. These processes were often implemented with relational databases at their core for quick query and data processing, and built as monolithic processing structures.

However, despite the increasing spread of techniques and applications, challenges still remain across the realm of information extraction, especially considering the growth of documents collected and eligible for further processing; in particular, these involve the need to automate document conversion and scraping, and the need for better methods of storage and retrieval [2]. Recent developments in information technology have allowed for the fulfillment of these needs, including (but not limited to) the use of serverless computing to distribute and scale conversion processes

across swaths of compute resources, and the emerging popularity of document-based databases allowing for the storage of associated and relevant metadata alongside collected documents.

2.2 Sentiment Analysis in Political Discourse

With the wealth of data provided and the polarizing opinions available, much sentiment analysis regarding political discourse has taken place using social media. Most of this analysis, however, has been conducted with regards to how social media shapes the political discourse. It was found that, rather than facilitating debate and changes in view through lively discussion between opposing sides, Twitter serves to effectively create an "echo chamber." Given how users can decide which voices to (literally) follow, they can, in effect, close themselves out from dissenting opinions and contrary voices necessary to provide a balanced view on the issue at hand and surround themselves with amplified versions of their own opinions [12]. Yet, at the same time, due to the democratic nature of the platform where prominent politicians are given as much as a platform as conventional constituents, Twitter can also serve as a neutral ground where users from wildly different social groups can trade ideas [10]. Another area of exploration brought about by the evolving platform was the spread of inaccurate (often politically-inclined) information and its effects on the discourse of the platform as a whole [14].

In addition, political figures and their words themselves have been the subjects of several studies, with one study using the tweets from candidates of the 2016 United States Presidential Elections to determine the attitude of politicians towards each other on the unique medium, and how much they interacted with each other [16].

Given that, there is still significant room for political discourse analysis, especially within the realms of true person-to-person interaction as it occurs within legislatures and across branches of government. The effect of legislative speeches and debate have not yet been investigated significantly as of yet—proving a significant opening for more focused research to pioneer.

2.3 Extracting Sentiment from Congress Transcripts

Beyond more simplistic political discussions accessible online, some governments make transcripts of their Congress floor debates publicly available online and often in machine-readable form. These are discussions of a higher level than what can normally be found in comments or blog posts. The language found in these debates is often more formal and precise, and can often be much more verbose. One particular study took interest in this kind of speech structure and attempted to create models for their analysis that went beyond techniques involving simple document-level sentiment-polarity classification [17]. Rather than assembling a whole document of speeches and running them through a classifier, the researchers recognized the dialogical nature of the speeches, and designed models accounting for the possibility of agreement between subjects in order to improve the accuracy of their classification. By identifying different-speaker agreements and factoring it into their analysis, they were able to move from 79.77% accuracy to almost 90% in their development set. It should be noted that they also used a same-speaker constraint—that is, the same label was attached to all the same person's speeches—to help them achieve such accuracy.

This study was able to spawn a whole host of other researches after they made their annotated data publicly available online. Their example is one of the driving forces behind this study's push to provide a public API for making its extracted data accessible to others. One such research leveraging the data of the aforementioned study tested the person-and-time-dependency of the classifiers trained in the data set by attempting to classify data from House speeches through training on speeches in the Senate and vice versa [5]. What they found is that some of the classifiers could indeed be generalizable to a fair degree of accuracy, but were largely time-dependent. This indicates that sentiment analysis in a political setting can be performed fairly reliably even when tested across both legs of the bicameral legislature.

Another study using the same data attempted to build upon the first research's idea of different-speaker agreements by examining *disagreements* instead [4]. What they found is that incorporating label-disagreement information into their models on top of the label-*agreement* improved the classification accuracy of the sentiment models even further than if they had just observed the agreement labeling. This opens the path for exploring additional dimensions beyond the speech contents itself in determining the polarity or predicted vote of speeches.

3 METHODOLOGY

In order to extract speech and vote data from legislative documents of the House—as was posed by the first question of this research—section 3.1 describes an integrated pipeline developed for converting structured documents to a digitized form. Within the context of the wider study, as shown in Figure 1, this requirement covers the 'Data Collection', 'Information Extraction', and 'API' portions as shown.

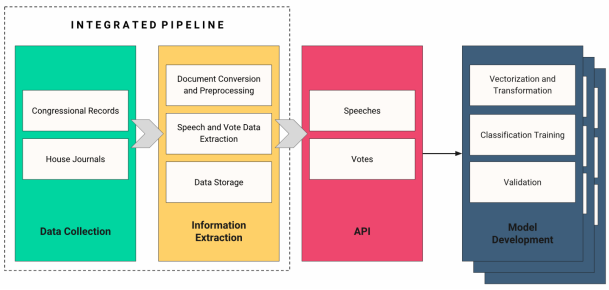

Figure 1: Framework of the wider study being conducted. The system described in this paper covers the "Data collection" and "Preprocessing" portions of the wider study.

To address the question on the creation of a classification model, section 3.2 of this study covers the development of a model to classify and predict a vote from the collected speech and vote data. More specifically, this model was used to determine either an affirmative or negative vote for representatives on bills given their speeches. As shown in Figure 1, this requirement covers the 'Model Development' portion of the study.

3.1 Integrated Pipeline

The 'Data Collection' and 'Information Extraction' portions were combined into an integrated pipeline so that the collected documents are automatically coursed through information extraction without further involvement on the part of the researchers. This integration was implemented using Amazon Web Services (AWS) Lambda platform, which allows for serverless function deployment, allowing for automated execution based on triggers such as new file additions, and effectively removing the need for server maintenance.

3.1.1 Data Collection. In order to collect the CRs and HJs available as PDF files on the official House website—and to easily source future days in Congress for processing and storage—a document scraper was created to automate the process of downloading the documents for further processing. The scraper used for data collection was created in Python with the use of the Beautiful Soup package. Beautiful Soup is an HTML parser that makes it possible to dig into the elements on a webpage and extract them for any specific use. In this case, the download links of all the CRs and HJs are isolated and compiled to allow for automatic batch downloading of all the files. The scraper was deployed as a serverless function on AWS Lambda, which facilitated daily execution (akin to a Cron job). Through this function, 1,161 Congressional Records and 728 House Journals were collected for further use in the study.

In order to organize and differentiate between the files for CRs and HJs, a naming system was devised so that each document type followed a different convention for the file name. The documents were stored in separate AWS Simple Storage Service (S3) buckets for further use and archiving purposes.

3.1.2 Document Conversion and Preprocessing. While the publicly-available documents contain all the necessary information for further data analysis, having it in PDF format does not allow for easy extraction. Furthermore, considering the structural complexity of a PDF document used in the formatting and design of the needed body text, a significant amount of unnecessary document features needs to be removed in order to get to the raw textual data.

A document conversion step was implemented to extract the useful textual elements from the document—and to exclude any unnecessary graphical flourishes, such as ornate line separators and multi-column text flows. This portion of the workflow was implemented in Node.JS to take advantage of the pdf2json package for text extraction from PDF documents, and in Python for text filtering and cleaning.

The document conversion step works to extract all written content from the page, including the body text, headers, and auxiliary information, into a format suitable for processing while still being readable by human eyes. This was accomplished through either the use of text extraction packages, if the PDF was created from another electronic source, or through optical character recognition otherwise. Lastly, the converted text was filtered for any occurrences of ornamental headers or any other unnecessary information before being passed on to the information extraction portion.

3.1.3 Speech and Vote Data Extraction. Much like the document conversion and preprocessing steps, the implementation of this particular step is specific to each document type. However, the overall process follows a general structure applicable to all documents.

The extraction process leverages the orderly structure of the document as present in its textual features. Figure 2 shows a sample structure of a typical document processed in this study. The CRs and HJs consistently demarcate each section of the document with headers, each describing different portions of sessions in Congress. Examples include the resumption and adjournment of sessions, the consideration of certain bills and measures, and the different types of voting to pass these bills. The predictable structure of the document was instrumental in the formulation of regular expressions that could parse through the volumes of text automatically and section them accordingly. Any further processing from this point onward is specific to the certain use case envisioned by the user.

CALL TO ORDER

At 10:00 a.m., Speaker Feliciano Belmonte Jr. called the session to order.

THE SPEAKER. The session is called to order.

NATIONAL ANTHEM

THE SPEAKER. All please rise for the singing of the National Anthem.

Everybody rose to sing the Philippine National Anthem.

THE SPEAKER. Please remain standing for the Invocation to be led by Cong. Lino S. Cayetano of the Second District of Taguig City.

INVOCATION

REP. CAYETANO. Let us bow our heads.

Almighty God, today, we embark on one of our most important functions here at the House of Representatives as we deliberate on the coming year's proposed national budget.

As wielders of the power of the purse, we pray for divine

Figure 2: Sample body text from Congressional Records.

In line with the language's emerging popularity in the field of natural language processing and in the wider data analysis scene, the preprocessing portions in this implementation were written in Python. Both the header removal and the section extraction portion pass the raw body text through a series of regular expressions (regexes) to separate the raw text into the sections based on the different headers found, and to filter out any unnecessary information, such as any remaining artifacts after the PDF-to-text conversion process. Formalized language was filtered in a similar way. Text from the section extraction portion passed through another series of speech pattern regexes to mark off the routine portions of debate.

3.1.4 Data Storage. In contrast to other studies involving information extraction that stored processed data in a relational (i.e., fully-structured) database [6, 17], the extracted votes and speeches are stored in a non-relational (i.e., semi-structured) database. In the case of this study, each speech segment is stored as a separate fragment, accompanied with the relevant metadata, including the specific document the speech can be found in, the bill the fragment is addressing, the speaker of the fragment, and, if available, the vote of the speaker on the bill. The semi-structured database was used as

different speeches and votes were found to have varying amounts of information (i.e., some speeches having votes while some did not). Instead of using a fully-structured database, which would necessitate adapting to the type of data with the biggest amount of information, the lack of definite schema of semi-structured databases makes the data storage process much easier, by removing the requirement of having to deal with default values or incompatible rows.

3.2 API

Given the wealth of information provided by the integrated pipeline in converting the CRs and HJs to a machine-readable format, an API was developed. Through the API, named the Floorreader API for how it "reads" discussions from the House floor, other applications may have access and use the converted plain-text and structured speech-vote data, and adapt it (as this study did with structuring data according to speeches and votes) to their own needs.

3.3 Speech Classification Model

To develop the model for classifying speeches according to the possible votes of representatives on bills, the speech and vote data collected from the 13th through the 16th Congress were paired together to facilitate a supervised learning methodology. This meant vote data were treated as target labels on which their corresponding speeches were trained. But because the data collected for use in the development of the model had a large imbalance between the number of samples available per each class, two primary models were developed for the task of classifying the speeches. Model A was developed to be representative of the actual dataset and trained with the imbalance included from the beginning, while Model B was trained on an equalized subset of the data. In order to feed the raw speech data into a classifier, the text was converted into numerical values using a term frequencyâĂŞinverse document frequency (tf–idf) vectorizer, which bases its conversion on how often a word appears in a document, prioritizing rarer words (such as “move” and “motion”), over more common ones (such as “Speaker” and “more”). Classification was then performed using Stochastic Gradient Descent, a class of linear support vector machine algorithm. The whole operation was tested and validated with ten-fold cross-validation to maximize the dataset and also to prevent over-fitting on any particular block of the data. Different features characterizing the speeches were added to the procedure afterwards to determine if they would have any bearing on the results in terms of performance.

The models were then evaluated using precision, recall, and F1-score. These performance metrics were chosen over the accuracy metric to gauge classification performance in order to lessen the effects of the large class imbalance in the collected data. Accuracy, when used as a performance measure in such situations, can be misleading because “classifiers” that consistently predict in one direction will be determined to be superior to classifiers that do otherwise as long as the data is skewed in the same direction. Compared to accuracy, the precision, recall, and F1-score metrics are less susceptible to this type of error.

4 RESULTS AND DISCUSSION

This chapter contains a discussion of the implementation of the pipeline and creation of the predictive model outlined in the study. In addition, the results of the predictive model are detailed and explained in this chapter.

4.1 Integrated Pipeline

The implementation of the framework described in this study was able to scrape and collect documents from the 13th Congress, beginning in 2006, through to the 16th Congress. The scraper was also found to function correctly with documents from the current-as-of-writing 17th Congress. This eleven-year span of document collection covered the extent of the electronic archives available on the congress.gov.ph website. In total, 112,736 speech units were collected in the debate of 752 bills, with 1,649 votes cast, with each of the speech segments labelled by the members’ respective votes if available. From data collection to storage, the entire workflow covered by the pipeline has been measured to take, on average, around five seconds to complete.

4.1.1 Data Collection. The web scraper for legislative documents was able to collect all the CRs and HJs made available on the House website, summing up to 1,161 CRs and 728 HJs. Some difficulties were found in consistently determining the new file names for documents from earlier Congresses, namely the 13th through the 15th Congresses, as they did not have as much information in their original file names as later Congresses had. The scraper was later written to resolve this issue by extracting the missing information from the contents of the documents themselves for these particular files.

4.1.2 Document Conversion and Preprocessing. For the document conversion and preprocessing step, the implementation was able to accurately extract speeches from CRs and vote records from HJs, and was also able to extract information from the Senate counterpart of the HJ, the Journal of the Senate, without substantial modification. This demonstrates the versatility of the framework in allowing for other similarly structured documents to be processed in a single implementation.

With the framework implementation, any headers included in future (or past) documents that are not currently covered by the implementation’s pattern-matching system would require code revisions to handle these cases. In addition, any new information introduced in the documents other than what was considered during the implementation would also be cause for a rewrite of the information extraction portion. For instance, from the 13th to 16th Congress, nominal voting records (i.e., votes of Congress members recorded) were omitted from the CRs, only to be introduced inline in the documents of the 17th Congress. This addition would necessitate rewriting some processing code to take into consideration these new pieces of textual information so as not to mix it up with other kinds of information extraction.

While some of these errors can be attributed to incomplete or incorrect implementation of the framework, it was found that there were many inconsistencies and errors in the documents themselves.

For instance, the spellings of "affirmative", "negative", and "abstention"—critical in parsing the votes of politicians of bills undergoing nominal voting—were often mangled throughout the HJs, with over 100 errors found across discussions of more than 166 bills from the 13th through the 16th Congresses. While these kinds of errors should be addressed at the level of authorship to improve the quality of documents for all users, any implementation of this framework should still be able to account for any errors that may arise out of negligence on the part of the data source. The study addressed these errors by correcting the typos in the source text file used for preprocessing.

Aside from typographical errors, the documents publicly accessible from the online archives of Congress are laid out in many different formats, with each successive Congress introducing its own minor changes. These modifications, while appearing as minor changes in the print version, were a cause for errors in text extraction.

Journals from the 13th and 16th Congresses provide significant examples for this phenomenon. Some of the first journals in the 13th Congress were typeset in a monospaced font, allowing for relatively easy text extraction. However, later journals were typeset in conventional serif and sans-serif fonts, which, while allowing for a more pleasurable reading experience, complicate information extraction and can cause inaccurate information to be returned.

For instance, the justified bill titles in recent journals may lead text extraction methods (text-based and OCR alike) to believe there are extra spaces within the titles themselves, instead of just being a feature of the typography (e.g., "AN ACT" becoming "A N A C T", as shown in Figure 3). This can also go the opposite way: on several occasions votes could not be parsed as the sans-serif font would appear to clump all names of politicians together.

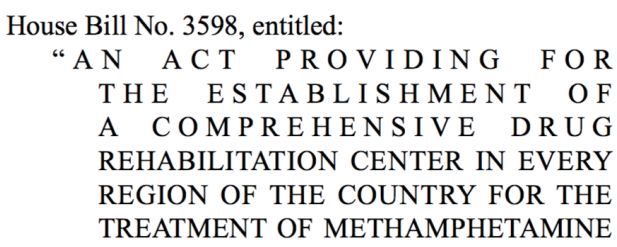

Figure 3: Inconsistent spacing in documents, as shown here, can result in inaccurate text extraction.

In essence, while the core structure of the document remained constant (i.e., section header in all caps, body text following, bounded by another section header) the content of the PDF was inconsistent. The study addressed these errors by fixing the spacing of text in the source text file used for preprocessing.

4.1.3 Speech and Vote Data Extraction. The implementation of the information extraction step follows the process outlined in Figure 4, and makes use of the inherent structure present in both CRs and HJs for efficient data extraction.

Firstly, the documents are sectioned according to the bill under discussion. Both CRs and HJs are structured according to the flow of sessions in the HoR, with each section signified by a unique header type. These headers, outlined in Table 1 for CRs and Table 2, mark the portions of the document which contain significant data for extraction.

Table 1: Relevant features from the Congressional Records and their purpose

Feature	Purpose
CONSIDERATION OF H.B. NO. []	Marks the start of debate on a certain bill
SUSPENSION OF CONSIDERATION OF H.B. NO. []	Marks the end of debate on a certain bill
All other sections	Removed from the document

Table 2: Relevant features from the House Journals and their purpose

Feature	Purpose
RESULT OF THE VOTING ON THE APPROVAL ON THIRD READING OF CERTAIN HOUSE BILLS APPROVAL ON THIRD READING OF HOUSE BILL APPROVAL ON THIRD READING OF CERTAIN HOUSE BILLS APPROVAL ON THIRD READING OF CERTAIN MEASURES APPROVAL ON THIRD READING OF CERTAIN BILLS	Marks sections for vote extraction
All other sections	Removed from the document

The speeches made by a Congress member—that is, everything they say during the period marked by the section headers—are grouped together. After filtering, the resulting speech data is labeled with the way each of the Congress members voted, in the affirmative or negative, for the respective bills. For example, if a certain Representative Dela Cruz were to make several speeches under the consideration of House Bill No. 2018, which he voted in the negative for, all speeches made by him regarding that bill would be tagged as being "negative". This is reasonable because, although his position on the bill might have changed throughout the discussion, the totality of his speeches should ultimately amount to the single encapsulating vote he cast at the end of the debate [17]. Therefore, his vote can be used to fairly represent the overall stance of the sum of his speeches.

4.1.4 Data Storage. While AWS S3 buckets were used for the storage of scraped documents and the text files resulting from the document preprocessing portion, the use of a full-fledged database system was necessary for the extracted vote and speech data. AWS' NoSQL-based DynamoDB service was used to store the processed and combined speech and vote data.

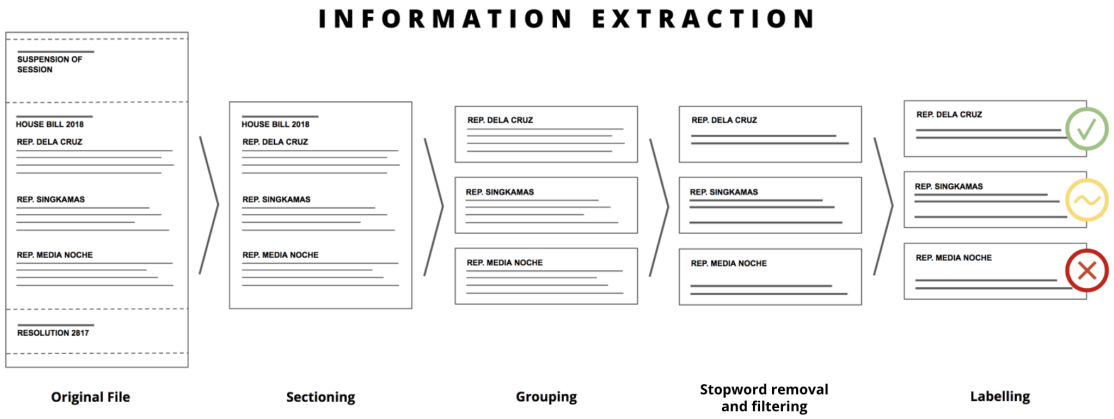

Figure 4: Diagram of information extraction workflow.

As DynamoDB tables do not require fixed schema—only needing primary sort and range keys for object sorting and automatic table partitioning, respectively—the study was able to store different amounts and types of data per speech, depending on how much information was available in both the CRs and HJs.

This format was also created in consideration of the storage needs of the DynamoDB database-as-a-service, part of the Amazon Web Services platform. DynamoDB's key-value-oriented structure fit well with how the study aims to serve the extracted data in the API. It is further reinforced by its tight integration with AWS Lambda, which allowed for swift reads and writes to the database, and its ability to scale for periods of high input (e.g., insertion of new data) and output (e.g., queries via the API). An example of the schema in action can be seen through how a speech with corresponding vote was stored in the manner specified in Figure 5.

```
{
    "BillID": "14#HB7122",
    "RecordBillSpeechID": "1R#30a#107",
    "MemberID": "14#LAGMAN",
    "Speech": "The point is completely...",
    "Vote": "affirmative"
}
```

Figure 5: Sample return values for an API call for a single speech and accompanying vote.

4.1.5 Integrated Pipeline Automation. As was mentioned in the methodology, the whole process described in sections 4.1.1 to 4.1.4 has been automated through the use of AWS Lambda—a serverless computing platform powered by Amazon Web Services. The platform afforded the ability to write each segment of the implementation in a specific environment catered to the segment's end goal, without having to maintain shared environments throughout the whole project.The Lambda platform allowed the study to use different languages and packages for each step of the information pipeline in isolation, without having to consider any cross-functional dependencies.

A graphical overview of the entire pipeline is provided in Figure 6. Note that each function (shown as the octagonal figures labelled with the Greek letter "lambda") corresponds to a different portion of the pipeline, indicated in the bold text next to each function.

Through AWS Lambda, once any new CR or HJ is uploaded to the House website, the entire data extraction procedure detailed in this study is triggered automatically. A fresh batch of uploads on their website can immediately and effortlessly be transformed into machine-readable speech and vote data, accessible via the Floorreader API and instantly ready for further processing on the part of any end user. The whole pipeline is put into operation without the need for any maintenance on the part of the researchers.

4.1.6 API Design. A working version of this implementation is available at *api.floorreader.ph*, with access granted upon request. The extracted data will be retrievable in JSON format through this URL. Speech data is structured as speeches made by representatives grouped by bill. The Floorreader API returns an array of bills discussed under single CRs structured as shown below. *This example and the examples that follow have been truncated in the interest of space.*

Vote data, as shown in Figure 7, is structured simply as a set of lists composed of the affirmative, negative, and abstention votes on a certain bill, with each representative's name included on the appropriate list corresponding to their vote. This data is available only for bills being voted on Third Reading. Similarly, speech data for a single speaker, also shown in Figure 7, is composed of a list containing all the speeches a certain member made on a single bill. It must be noted that the speech data is not returned with the vote data even if it is present in the database.

4.1.7 Data Structuring for Classification Model. As the wider project focused on relating representatives' speeches in bill debates to their corresponding votes for bills on their Third Reading, data from both the internal CR and HJ APIs were condensed into a structure that can be directly used to correlate the two. Effectively,

, ,

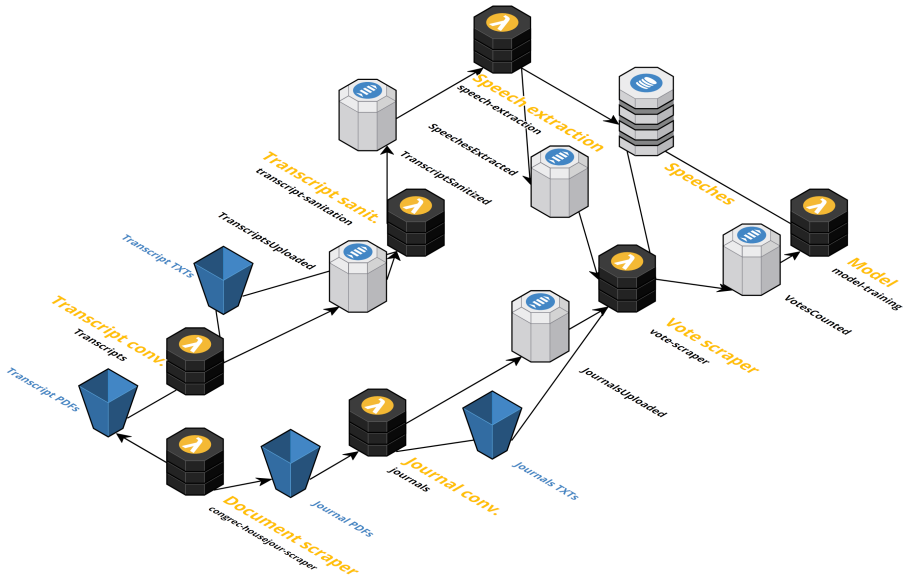

Figure 6: Overall structure of the information extraction pipeline as deployed on AWS Lambda as a series of connected serverless functions, with accompanying storage buckets for documents and database for extracted speech and vote data.

vote and speech data were joined on a specific Bill and Member ID to obtain a document as in Figure 8.

This specific format groups speeches by congress number, the house bill being addressed and the representative making the remarks. Their eventual vote (or lack thereof) on the bill's third and final reading in the House, if the bill has made it to that stage, is included in the data.

4.2 Model Development

4.2.1 Speech Vectorization Process. The raw speeches were first tokenized through a count vectorizer, which essentially converted textual data (word occurrences) into numerical feature vectors. These were then put through a tf–idf transformer to intelligently account for the presence of words against the length and breadth of the speeches. This was done to normalize the average count of words in longer speeches as compared to those in shorter ones (term frequency) and to decrease the relative importance of words occurring in many speeches as opposed to those occurring in only a few (inverse document frequency). Doing so would help reduce the impact of words falling under parliamentary language, which are recurring but unnecessary. From there, classification was performed with Stochastic Gradient Descent (SGD), a type of linear support vector machine. SGD was chosen for its time efficiency and performance over other classification algorithms commonly used in textual analysis.

4.2.2 Performance of Model A. Table 3 shows the confusion matrix of Model A, which was trained on the complete dataset of 1,645 speeches. Abstain, affirmative, and negative votes were all components of the data comprising this model. The returned performance scores of the model can be seen in Table 4.

Table 3: Confusion matrix of Model A.

		Predicted			
		Abstain	**Affirmative**	**Negative**	**Total**
Actual	**Abstain**	0	33	0	33
	Affirmative	1	1484	0	1485
	Negative	0	127	0	127
	Total	1	1644	0	1645

Table 4: Performance scores of Model A.

	Precision	**Recall**	**F1-score**	**Support**
Abstain	0.00	0.00	0.00	33
Affirmative	0.90	1.00	0.95	1485
Negative	0.00	0.00	0.00	127
Avg / Total	0.81	0.90	0.86	1645

The raw averages of the performance scores of Model A render good percentages at 81% average precision, 90% recall, and 86% F1-score. These numbers, however, cannot be interpreted without first taking into account the considerably uneven distribution of classes. Only 8% (127) of the total speeches produced by the extraction pipeline had negative votes. Meanwhile, 90% were affirmative (1485) and 2% abstain (33). Because of this, the model was trained with a large skew towards classifying in the affirmative. This had all but one of the speeches being categorized by the model in the affirmative as well, resulting in very high accuracy, but no speech being classified correctly in the abstain or negative classes.

4.2.3 Performance of Model B. The confusion matrix of Model B, under which the uneven vote distribution was equalized, can be

```
{
    "BillID": "14#HB6215",
    "Status": "Passed on Third Reading",
    "Votes": {
        "affirmative": [
            "14#A.ABAYA",
            "14#M.DIMAPORO",
            ...
        ],
        "negative": [
            "14#A.DIMATULAC",
            "14#G.MACATOL",
            ...
        ],
        "abstention: []
    }
}

{
    "BillID": "14#HB6215",
    "Debate": [
        {
            "MemberID": "14#M.DIMAPORO",
            "Speeches": [
                "The beneficiaries of..."
                "Yes, including communities...",
                "Barangays, municipalities,...",
                "The identification of...",
                ...
            ]
        }, ...
    ]
}
```


Figure 7: Sample return values for a Floorreader API call for all speeches and votes on a single arbitrary bill that has passed Third Reading.


```
{
    "BillID": "14#HB6215",
    "MemberID": "14#M.DIMAPORO",
    "Speeches": [
        "The beneficiaries of..."
        "Yes, including communities...",
        "Barangays, municipalities,...",
        "The identification of...",
        ...
    ],
    "Vote": "affirmative"
}
```


Figure 8: Sample values returned for classifier training.

seen in Table 5. Meanwhile, the performance of Model B can be seen in Table 6.

Table 5: Confusion matrix of Model B.

		Predicted		
		Affirmative	**Negative**	**Total**
Actual	**Affirmative**	96	31	127
	Negative	9	118	127
	Total	105	149	254

Table 6: Performances scores of Model B.

	Precision	**Recall**	**F1-score**	**Support**
Affirmative	0.91	0.76	0.83	127
Negative	0.79	0.93	0.86	127
Avg / Total	0.85	0.84	0.84	254

Given the 127 speeches in the corpus tagged with a negative vote, a random set of 127 speeches was also taken from the affirmative set to create a balanced cluster on which to train. The 33 speeches tagged as "abstain" were discarded. Speech tokenization for this model was also slightly adjusted by disregarding a list of basic English and Filipino stop words and also by incorporating bigrams into the feature vectorization on top of just having the unigrams initially.

Model B produced far more promising results. Although its average recall and F1-score dipped slightly, the model's average precision went up to 85%. More importantly, it was able to classify speeches into the negative class with scores of 79%, 93%, and 86% for precision, recall, and F1-score respectively, compared to the zeroes across the board for these three metrics for Model A.

4.2.4 Prominent Classification Features. Because Model B was able to distinguish between speeches resulting in affirmative and negative votes, it became useful to analyze the top feature coefficients of the classification. These are measures of the importance of each feature, which in this case are words, in the classification of a speech towards a specific category. The higher the absolute value of the coefficient, the stronger it correlates with its particular class. Shown in Figure 9 are the top 20 n-gram features for speeches classified in the affirmative and negative classes. Positive values on the right are those belonging to the affirmative class while negative values on the left are for the negative class.

Noteworthy in the chart for affirmative features was the large occurrence of words falling under parliamentary language—mainly, the very high coefficient of the unigram *move*. This would make it appear as though representatives voting in the affirmative are not much interested in actual debate, but more with simply moving the proceedings forward.

4.2.5 Experimentation with the Dataset and Model. To try to improve the performance of the Model B, different experiments were conducted to add more features to aid the classification. Titles of the bills and their corresponding category (formally known as the bill's "referral"), for example, were scraped from the Congress

, ,

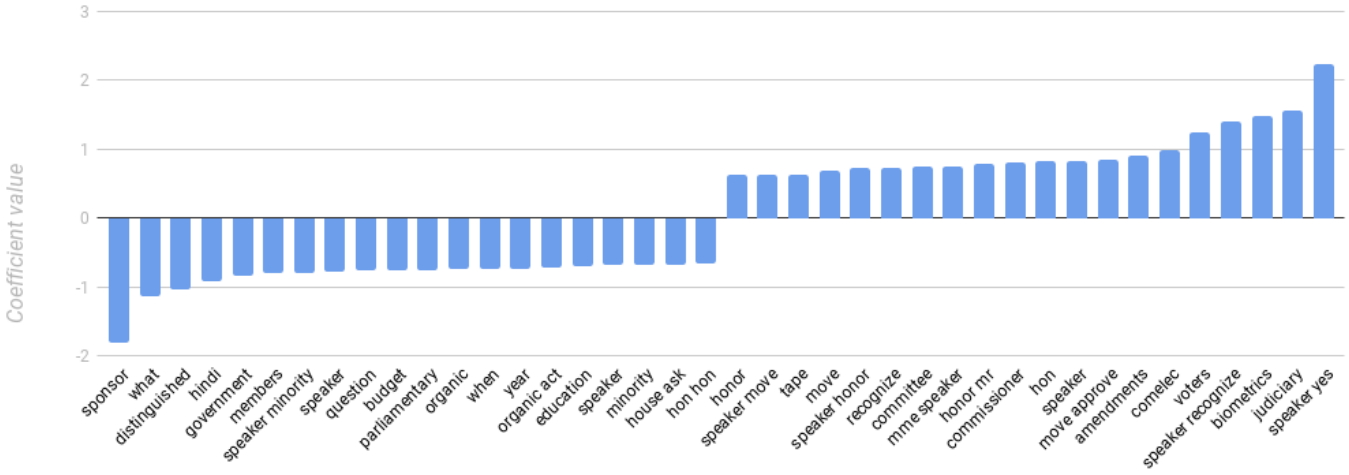

Figure 9: Prominent n-gram features in the affirmative and negative classes.

website to serve as supplementary features on top of the words found in the speeches. However, as some bills documented in the records did not appear on the website, the corpus of 127 speeches for each class was constrained even further to those speeches having a bill title and referral. This about halved the whole cluster for training and validating and so did as much to hurt performance as it did help it. Other attempts to add more dimensions did not appreciably affect performance.

4.2.6 Political Analysis of Uneven Vote Distribution. The heavy skew towards affirmative votes in the passing of bills can be taken to mean many things. For example, it is possible that hotly-contested bills rarely ever reach the third reading and are shelved. When these bills are reintroduced, they may have been changed to address the concerns of the representatives that had initially voted in the negative. It is also possible that most bills going through the lower house of Congress do not warrant much debate in the first place–bills having inconsequential effects (e.g., the renaming of schools or the declaration of public holidays). Or perhaps even—quite unfortunately—that members of the House of Representatives simply pass bills on order from those in higher seats of power.

5 CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

In summary, this paper was able to outline and implement a system for the extraction of relevant text from floor debate records of the House of Representatives of the Philippines, while at the same time laying the foundations for a framework to extract information from a wider set of documents throughout government. The framework involved breaking down the structure of the PDF files and the text itself to get the raw information, while using textual features to section the information into usable data that can be structured for further analysis.

The implemented pipeline was able to accurately process and convert almost two thousand records and journals from the House of Representatives, from the 13th up to the 16th Congress and beyond with minimal modification, and was open to processing documents from other branches of government, such as congressional records and journals from the Senate.

The study was also able to provide an API for retrieving speech and vote data from the above corpus. It is hoped that the information made available through the Floorreader API is used to create applications that help make analyses of political debates more relevant, and make legislative data itself more accessible to the average Filipino citizen. For instance, the speeches made available here could be used without further analysis to visualize the activity of politicians in debates, showing how often a politician speaks regarding certain topics or bill referrals. On another note, the vote data can be coupled with more information on the bills themselves (such as their titles, referrals, and actual body text) to provide readers with better insights as to how politicians vote on salient measures, and whether their representatives really do represent their views, hopefully leading voters to make more informed decisions.

The study was also able to develop a model for classifying the speeches of representatives made in the debate of bills as being either those that would lead to an affirmative vote for or a negative vote against a bill. The model achieved solid performance scores of 85% in precision and 84% in recall and F1-score on the extracted speech and vote corpus, as calculated by the scikit-learn Python package. However, this was managed only after equalizing the dataset to account for the greatly uneven distribution of negative and positive votes. A model trained on a real-world distribution only managed to classify speeches in the affirmative and will prove almost useless.

5.2 Recommendations

Given how this study focused on the development of the framework and an initial implementation of the information extraction pipeline and development of the predictive model, there are many

opportunities for further research. Possible improvements leading to further studies are detailed in this section.

5.2.1 Integrated Pipeline. Modifications to the integrated pipeline can be made on any portion of the framework to enhance performance or to improve the accuracy of the extracted information.

Print Document Collection: A limitation stated earlier in the study concerned the lack of digitized versions of CRs and HJs from congresses earlier than the 11th Congress. Existing technologies such as optical character recognition can be combined with mechanical technologies for handling the paper documents to create a fully-automated system for creating digital versions of documents. Interested researchers could use or build upon existing mechanical book scanners, and work in tandem with the Library of the House of Representatives to adapt these systems to the different shapes and sizes of the CRs and HJs in their archives.

Expanded Information Extraction: While this study focused mainly on the extraction of speech and vote data from the CRs and the HJs, there remains a lot of data that can be retrieved from these documents. These include the attendance of the representatives, the bill names and their referrals (so as to remove the need to extract them from the HoR website), correspondence with the Senate, and privilege speeches that happen outside the bounds of conventional debate. This information can be used not only to potentially improve the model, but also to expand the information provided in the Floorreader API, and make this data available for other researchers to use.

Improved Document Preprocessing and Section Extraction:

While the regex-based implementation of the pipeline has sufficed for the purposes of the wider investigation, it is clear that a better method involving the graphical organization of the documents in the analysis would greatly enhance the efficiency and reliability of the pipeline.

With the pipeline implementation, any headers included in future (or past) documents that are not currently covered by the implementation's pattern-matching system would require code revisions to handle these cases. In addition, any new information introduced in the documents other than what was considered during the implementation would also be cause for a rewrite of the information extraction portion. For instance, from the 13th to 16th Congress, nominal voting records (i.e., votes of Congress members recorded) were omitted from the CRs, only to be introduced inline in the documents of the 17th Congress. This addition would necessitate rewriting some processing code to take into consideration these new pieces of textual information so as not to mix it up with other kinds of information extraction.

A more optimized implementation would take advantage of computer vision and artificial intelligence to recognize and detect useful elements of documents—instead of pushing all extracted text through a series of regexes, as was done in the implementation of the framework in this paper, filtering and segmentation can be done based on the graphical organization of the text, and would greatly reduce the number of assumptions made in the structure of the document.

Adding in machine learning techniques would also open up the framework to not only new documents (so as to build a fully functioning system capable of extracting relevant text from all levels of government), but also modifications of existing documents. New formats or additions (such as the inclusion of vote information in the CRs starting in the current-as-of-writing 17th Congress, or the induction of new House members, as shown in Figure 10) would easily be recognized and organized in a manner suiting the information, without the need to manually program for each possible configuration of the document, especially formats created for one-off or infrequent occurrences, such as the listing of House members at the start of each new Congress.

SBP (Serbisyo sa Bayan Party)
Hon. Ricardo T. Belmonte Jr.

YACAP (You Against Corruption And Poverty)
Hon. Benjur B. Lopez Jr.

THE DEPUTY SECRETARY GENERAL (Atty. Adasa). Mme. Presiding Officer, the roll call shows that 285 Members responded to the call.

THE PRESIDING OFFICER (Atty. Barua-Yap). There being 285 Members who responded to the call, the Chair declares the presence of a quorum.

Figure 10: Unique formats often emerge in these documents that can easily be detected with more advanced algorithms.

5.2.2 Stance Extraction Model. Although the study was not able to expand dimensionality useful to the improvement of the model beyond the features extracted from speeches, adding features for the specific voting member and authors of the bill being voted on (or their respective partylists) can be used to expose the partylines along which representatives vote. Such information is available on the House website and can easily be scraped just as the legislative documents were for this study.

To further verify the efficacy of the model, it is also recommended that it be tested on a yet-to-be-seen corpus of data—the 17th Congress, which is still ongoing as of writing. Since it has not concluded, it was not included in the initial testing and validation of the classification model. However, it may serve as an additional verification step to further legitimize the applicability of the model.

ACKNOWLEDGMENTS

The authors would like to express their sincere thanks to Dr. Marlene M. De Leon of the Social Computing Science Laboratory (SCSL) of the Ateneo de Manila University for her invaluable guidance throughout the development of this thesis, from the initial conception to the final output. This project would not have been possible without her ever-present support.

The authors would like to thank Dr. Andrei Coronel of the Department of Information Systems and Computer Science and Dr. Maria Regina Justina Estuar of the SCSL for their assistance in the development of this framework and implementation, and for their guidance in helping form the wider project of analyzing political discourse in the Philippine House of Representatives.

The authors would also like to extend their gratitude to Beatrice Adajar for her assistance in developing one of the scrapers used for extracting information from the House of Representatives website.

Lastly, the authors would like to thank the librarians of the Library of the Senate of the Philippines for their assistance in navigating the journals and records of the Senate, and for helping determine the feasibility of the study with regards to the resources available for that branch of Congress.

REFERENCES

[1] 2014. Open Data Philippines Action Plan 2014-2016. (2014). http://data.gov.ph/sites/default/files/Open%20Data%20Philippines%20Action%20Plan%202014-2016.pdf

[2] Flora Amato, Francesco Colace, Luca Greco, Vincenzo Moscato, and Antonio Picariello. [n. d.]. Multimedia data integration and processing for E-government. ([n. d.]).

[3] F Amato, A Mazzeo, V Moscato, and A Picariello. 2009. Information extraction from multimedia documents for e-government applications. In *Information Systems: People, Organizations, Institutions, and Technologies*. Springer, 101–108.

[4] Mohit Bansal, Claire Cardie, and Lillian Lee. 2008. The Power Of Negative Thinking: Exploiting Label Disagreement In The Min-Cut Classification Framework. In *Proceedings of COLING: Companion volume: Posters*. 15–18.

[5] Stefan Kaufmann Bei Yu and Daniel Diermeier. 2007. Ideology Classifiers for Political Speech. (2007).

[6] Allan Borra, Charibeth Cheng, Rachel E. O. Roxas, and Sherwin Ona. 2011. Information Extraction and Opinion Organization for an e-Legislation Framework for the Philippine Senate. In *Proceedings of the 2011 Conference on Human Language Technology for Development*. 196–204.

[7] T. T. Cheng, J. L. Cua, M. D. Tan, K. G. Yao, and R. E. Roxas. 2009. Information extraction from legal documents. In *2009 Eighth International Symposium on Natural Language Processing*. 157–162. https://doi.org/10.1109/SNLP.2009.5340925

[8] Jim Cowie and Wendy Lehnert. 1996. Information extraction. *Commun. ACM* 39, 1 (1996), 80–91.

[9] Matthew Gentzkow and Jesse M Shapiro. 2010. What drives media slant? Evidence from US daily newspapers. *Econometrica* 78, 1 (2010), 35–71.

[10] D King, Ramirez-Cano D, F Greaves, I Vlaev, S Beales, and A Darzi. 2013. Twitter and the health reforms in the English National Health Service. *Health Pollicy* 110 (2013), 291–297.

[11] Chih Hao Ku, Alicia Iriberri, and Gondy Leroy. 2008. Natural language processing and e-Government: crime information extraction from heterogeneous data sources. In *Proceedings of the 2008 international conference on Digital government research*. Digital Government Society of North America, 162–170.

[12] D LycariÃčo and M.A dos Santos. 2016. Bridging semantic and social network analyses: the case of the hashtag precisamosfalarsobreaborto (we need to talk about abortion) on Twitter. *Information, Communication Society* 20, 3 (2016), 368–385.

[13] Ann Macintosh. 2008. E-democracy and e-participation research in Europe. In *Digital Government*. Springer, 85–102.

[14] Jacob Ratkiewicz, Michael Conover, Mark Meiss, Bruno Gonçalves, Snehal Patil, Alessandro Flammini, and Filippo Menczer. 2011. Truthy: mapping the spread of astroturf in microblog streams. In *Proceedings of the 20th international conference companion on World wide web*. ACM, 249–252.

[15] Citizen-Voters Education Secretariat. 2005. Workshop Planning on Citizen Voters' Education. (2005). Institute for Electoral and Political Reform, Retrieved October 31, 2017 from http://www.iper.org.ph/CER/citizenvoterseducation/resources/acworkplanning.html.

[16] Q Simms. 2016. Twitter predicted the results of the Presidential Primaries. Could it predict the general election, too? (2016). ParseHub, Retrieved October 1, 2017 from https://blog.parsehub.com/what-27000-tweets-can-tell-you-about-the-presidential-primaries/.

[17] Matt Thomas, Bo Pang, and Lillian Lee. 2006. Get Out The Vote: Determining Support Or Opposition From Congressional Floor-Debate Transcripts. In *Proceedings of EMNLP*. 327–335.