Applying Semantics for the Analysis of Political Journalism

Avery Iorio, Kirk Olkowski, Nathaniel Adair

{iorioa, olkowk, adairn} @rpi.edu

Abstract

The increase in access to online news media has resulted in increased pressure for journalists and news publishers to cater to a particular viewer base often resulting in polarized coverage. In this paper we aim to build a system that allows users to analyze political journalism articles by leveraging semantics to connect these articles, journalists and publishers to the events, people, and topics being covered. By incorporating tens of thousands of articles from *The New York Times* and *Fox News* as well as federal election data we attempt to bridge these gaps in article keyword tags and allow for the comparison of media coverage across time and different news outlets. This required extensive conceptual modeling of elections, candidacy roles, and topic subclassing to infer relationships not explicitly described by article keywords. Though challenges like inconsistent tagging and the linking of elections to their mentioning in articles remain, our work highlights the potential for how semantic tools might improve understanding of journalism trends and reduce media bias.

1 Introduction

With the proliferation of information technology, people have greater access to news media than ever before. This explosion in accessible perspectives, combined with the greater pressure for journalists and reporters to distinguish themselves from the ever increasing flood of information, has led to increased polarization. We provide an ontology-enabled semantic system which allows the user to parse and understand journalistic coverage in light of the biases of the journalists and news agencies. We aim to empower users to develop a robust understanding of the interaction between the events, the news, and the biases of those reporting these events. News agencies often use differing terminology to discuss the same issues, politicians, and events. For instance, a right-leaning agency might run a story tagged with immigration, whereas the same story is covered in a left-leaning agency under human-rights. This type of tagging often reflects the terms used within the differing segments of the population, leading each side toward their preferred echo chamber. By leveraging semantic technologies, we can connect and disambiguate these differences, allowing our users to connect politicians, legislation, political parties, elections, to the commentary across the journalistic spectrum. Note that we do not attempt to establish bias in any objective sense, and instead compare the relative bias between articles, journalists, news agencies, etc.

2 Related Work

Due to the existence of news aggregators and the increasing popularity of historical news archives, there has been a substantial effort to standardize the syndication of online media (Bødker 2018). This goal has been partially met by the wide-spread adoption of Real Simple Syndication (RSS) Feeds which provides an XML based standard for media outlets to share newly published articles. The primary issue with RSS feeds is that despite adopting terminology from the Dublin Core Metadata Terms for describing web resources, there is relatively little standardization regarding article keywords or tags. These tags typically consist of around a dozen short strings capturing the most important topics, people, or events covered by the article. While many major media outlets have an internal ontology team that curates a list of article tags, there is no standardized set of tags used by most media outlets. Outlets like The New York Times use a combination of rule based categorization alongside human checking, but still have inconsistencies and bias in which tags are chosen (see section: automatic tag generation).

When it comes to having a shared vocabulary for representing elections, the Ontology of Election (Nigerian Election Ontology) and POWER (Portuguese Election Ontology) have thoroughly modeled aspects of the election process specific to each nation (Moreira et al. 2011). However, these ontologies were created primarily to model political offices and were not designed with the goal of tracking elections and candidates across time or linking candidates to political articles mentioning them. More general efforts have been made by outlets like the BBC with their family of news related ontologies. However, these ontologies suffer from few terms and virtually no semantic capabilities for augmenting article tags or linking people to specific events or topics.

The SNaP Ontologies represent the assets, events, and entities which appear in news coverage. These ontologies

cast a wide net in capturing representations of the various newsworthy events, and their contents. These representations are designed to capture various relationships in generality, taking the view that it is better to capture general relations rather than get bogged down in the specific details of such relationships. Given our ontology's narrower focus on the modeling of specifically political events as they relate to the news, we require more precisely specified relationships in order to leverage the semantic power which ontology-based systems are designed to harness.

The BBC Ontologies form the cohesive, current basis for the BBC's linked data platform, allowing representation of the world, BBC content, and data management, storage and sharing within their platform. These include a Politics Ontology and a Journalism Ontology. The BBC Ontologies do not provide a platform toward allowing the comparison and comparative evaluation of the differing journalistic outputs.

The W3C Ontology for Media Resources captures the format, content, contributors and other specifications of various media resources, divided between both the concept of a work, e.g. Hamlet, and a representation of the work, e.g. a specific MPEG-4 encoding of the English version of Hamlet. This ontology provides a more detailed representation of the actual formats of various media, but does not represent the actual events, if any, which are related to the works themselves, much less in a semantically tractable manner.

News Hunter is the most similar in use and scope to what we demonstrate with the Political Journalism Ontology, differing in two primary areas. News Hunter is designed to collate and tag data harvested from social media feeds in order to present potentially relevant information to a journalist as a situation is unfolding, whereas the Political Journalism Ontology is designed to enable user analysis of existing journalistic output. News Hunter is designed with the intention of aiding journalists in discovering relevant data toward their desired spin or slant on the situation. Conversely, we seek to reveal such biases to the reader.

3 Technical Approach

To address the deficiencies in how political journalism articles are currently represented we sought to patch two specific gaps. The first gap was to use election data to infer facts about current political office holders and the structure of elections. The second was to use this election data to augment and enrich the keywords used to tag articles.

3.1 Data Scraping

To validate our approach to analyzing and categorizing political journalism articles we first needed to collect data from two primary sources.

3.1.1 Election Data

The United States government publishes data on the details and outcomes of past federal elections that can be downloaded as CSV files from

https://www.usa.gov/election-results

While the election data exists in a relatively clean format the difficulties in ingesting this data to support our ontology's semantics was more difficult. For example, Bernie Sanders is a candidate who does not run as a member of the Republican or Democratic political parties but still caucuses the democrats.

3.1.2 New York Times Article Archive

The United States government The first of two media outlets that gathered articles from was *The New York Times*. This was an obvious choice as *The New York Times* is a widely read national newspaper with a left leaning bias that supports a robust API for accessing archived articles. To process this data we used the requests library in Python to call the NYT Archive API and were able to pull over 140,000 articles from 2020 to 2022. After filtering out articles from news desks unrelated to political journalism (e.g. sports, entertainment, etc), articles without at least one individual person author, and articles without title abstracts we were left with 22,795 articles. Finally, we created a script using the RDFlib module in Python to create an RDF with individuals for each unique article, author, date, and publisher.

The NYT also supports a lightweight taxonomy for classifying article keywords and has an internal system for creating new keywords as novel terms come into the public discourse.

Concept Type	concept_type	specific_concept_name
Descriptor	nytd_des	Absenteeism
Location	nytd_geo	Acapulco (Mexico)
Organization	nytd_org	3M Company
Person	nytd_per	Abbas, Mahmoud
Public Company	nytd_porg	
Title	nytd_ttl	The Joy of Painting (TV Program)
Topic	nytd_topic	

3.1.3 Fox News Article Scraping

The second media outlet we chose to use articles from was *Fox News.* This seemed like a natural choice as *Fox News* is a massive media outlet with a strong right leaning bias, has tens of thousands of articles published, and provided a good counterpart to the NYT. However, neither *Fox News* nor any right leaning news outlet as far as we could find has a publically available API for accessing archived articles. To solve this issue we decided to scrape several *Fox News* RSS feeds as the resulting JSON data from these

feeds are somewhat similar to the NYT Archive API responses. The next problem was that RSS feeds typically have articles from only a week or two before the present date. We used the Internet Archive's WayBackMachine which consists of historical snapshots of the internet at regular intervals to access past RSS feeds (https://web.archive.org/).

This approach required a significant delay (~20 seconds) between calls to the WayBackMachine API to ethically comply with its rate limits and avoid overloading the server. Additionally we had to deduplicate articles that persisted in the RSS feed across snapshots. In the end this allowed us to scrape over 38,000 unique *Fox News* articles from July 2022 to August 2024.

The *Fox News* RSS feeds have a slightly more in depth taxonomy for classifying keywords, but the generation and assignment of these keywords is not particularly standardized and leaves room for the use of semantics to infer additional keywords to describe an article.

```
:"foxnews.com/metadata/prism.channel">fnc</category>
:"foxnews.com/metadata/dc.source">Fox News</category>
:"foxnews.com/taxonomy">fox-news/politics</category>
:"foxnews.com/taxonomy">fox-news/us/us-regions/northeast/pennsylvania</c
:"foxnews.com/taxonomy">fox-news/politics/elections/voter-fraud-concerns
:"foxnews.com/taxonomy">fox-news/politics/voting</category>
:"foxnews.com/taxonomy">fox-news/politics/elections/presidential</catego
:"foxnews.com/taxonomy">fox-news/us/crime</category>
:"foxnews.com/section-path">fox-news/politics</category>
:"foxnews.com/content-type">article</category>
Dec 2024 14:06:33 -0500</pubDate>
```

Metrics

Axiom	156,202
Logical axiom count	102,006
Declaration axioms count	503
Class count	240
Object property count	183
Data property count	27
Individual count	25,028
Annotation Property count	38

Class axioms

SubClassOf	372
EquivalentClasses	34
DisjointClasses	15
GCI count	0
Hidden GCI Count	32

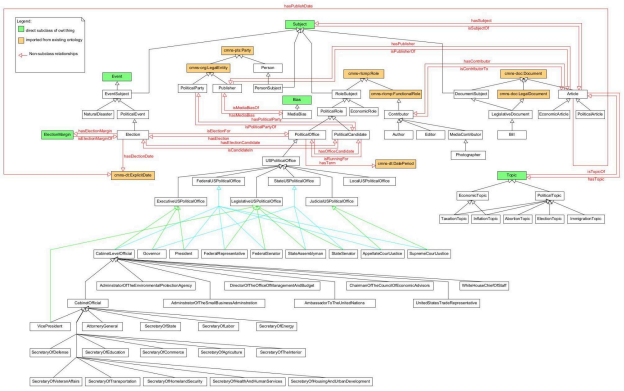
3.2 Use Case

We aim to facilitate user's ability to discover bias and trends as relates to specific journalists and news outlets and their coverage of particular events and persons. The primary difficulty lies with the diverse terminology which is used across the various news outlets to describe and comment on even the same events. Even coverage by the same news outlet will drift over time periods as short as a year or two. Semantic technologies, specifically ontology-enabled knowledge graphs, address the issue, through the association of coverage to events via tags by understanding and relating the semantic content of both the provided and system generated tags. For instance, Fox News might tag an article about Hamas bombing a hospital with a terrorism tag, whilst an article in the New York Times might tag an article covering the same event with humanitarian crisis and Hamas. The differing biases in such coverage would obscure the link between such articles without the understanding that Hamas is a terrorist organization, and that these articles in fact refer to the same event.

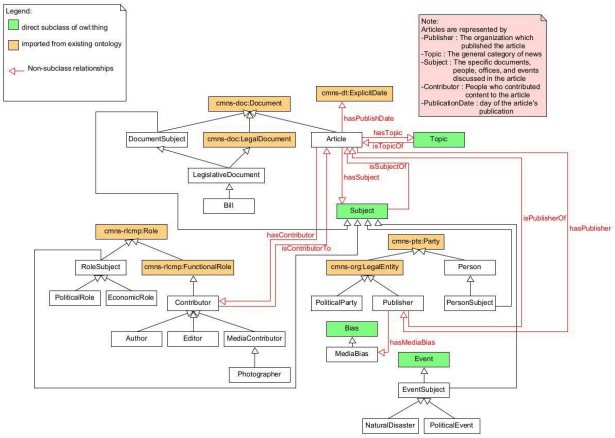
Semantic technologies allow the integration of data external to the articles themselves in order to provide a backdrop of relevant information over which to assess and process article content. In particular, the system can aggregate election data from official government sources, in this case the US Federal Election Commission. This provides a rich understanding of the political landscape, lays the foundation for the capture of indirect and obtuse references to existent political figures.

3.3 Conceptual Model Diagrams

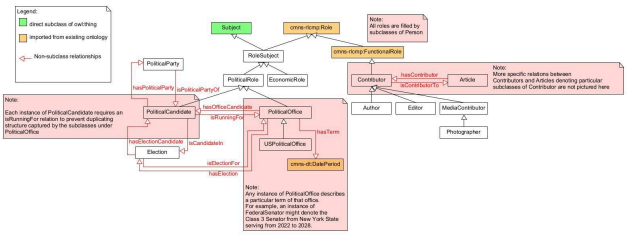
WeThe following conceptual model diagrams detail the classes and high level object properties of the ontology.



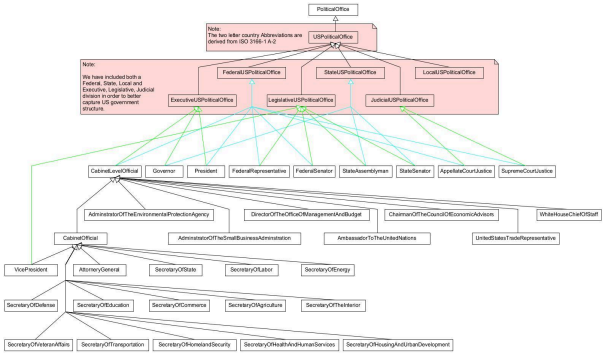
At the top level, we are concerned with the representation of articles, and their contributors, subjects, topics, and publishers. A contributor represents a person who has contributed to the content of a given article, usually an author or editor, though other types of contributors, such as photographers, exist. A subject represents a person, office, event, or thing which an article is written about. This includes our detailed representations of political offices, political candidacy, and elections. A topic represents a general category within which an article would fall. Examples include political or economic, inflation, and immigration. Topics can often be derived from subjects, but allow for more general categorizations. Publisher captures the specific outlet which published a given article.



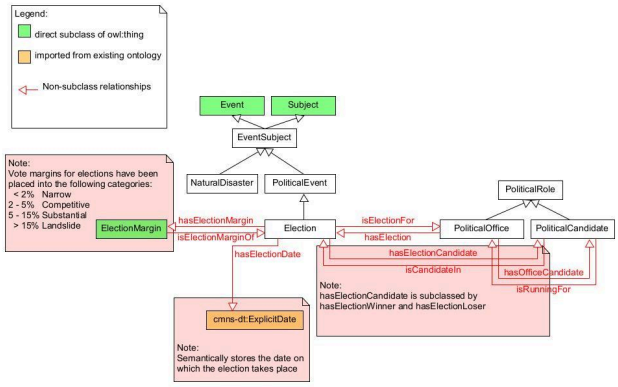
Of these top level classes, subjects are the most complicated, due to the variety of possible subjects and the complexity of representing such subjects in a manner to capture sufficient semantic content allow for the utilization of the semantic reasoning capabilities of the system. Given that, subjects are divided into four categories: person, document, role, and event. A person represents a real life person, such as Donald Trump or Kamala Harris. A document might represent a bill or other important piece of legislation. Roles in this instance capture political offices. To illustrate the necessity of this arrangement, let us consider Donald Trump, who, as of the writing of this paper, is a Former President, a recent Presidential Candidate, and the President-Elect. Utilization of the role structure, as defined in the Commons Roles and Compositions ontology allow the maintenance of a continuity of personhood, while allowing for the multitude of roles that a given person may fulfill at any given time.



The political focus of our ontology required the creation of a fairly detailed representation of the US political structure at the federal and state levels.



Events are a general category designed to capture the ‘things which happen’, such as elections. Any given election has a list of candidates, an office which the election is for, and a margin of victory.



4 Evaluation

4.1 Competency Questions

Our ontology is evaluated on its ability to answer some core competency questions. These questions were generated as example questions that the system should be able to answer if prompted. The competency questions are also used to guide the scope of the project.

These competency questions are as follows:

1. What federal senate elections in 2022 that had a narrow victory margin for a Republican candidate were mentioned in articles published by a right-wing news outlet?

   Example Answer: Senator Lisa Murkowski (R-AK) and Senator Ron Johnson (R-WI) won their 2022 senatorial elections by a narrow margin and were mentioned in articles published by Fox News.

   Reasoning: We will query for 2022 elections which have winners who are associated with the Republican Party, and check to see which of those elections were won by a narrow margin. Then we verify that those elections were covered by articles published by right or lean-right publishers, in this case Fox News.

2. Which journalists wrote articles about Senator Bernie Sanders in both the New York Times and Fox News from 2020–2022?

   Example Answer: There are no journalists who have written articles mentioning Bernie Sanders for both the New York Times and Fox News.

   Reasoning: We will need to create two queries looking at various articles during 2020–2022 which mention Bernie Sanders from the New York Times and Fox News respectively. We will

need to see which journalist wrote each article and which of these journalists are present on both lists.

3. What articles have the New York Times and Fox News published about Democratic candidates in relation to the economy in 2022?

   Example Answer: In 2022, Fox news published [Article 1] and [Article 2] about Democratic Candidates and the economy, while the New York Times published [Article 3] and [Article 4].

   Reasoning: We will query for articles which mention Democratic Candidates as well as an economic topic. These articles will be filtered by publisher for Fox News and the New York Times and by date for 2022.

4. What journalists wrote an article published by the New York Times in 2022 mentioning a Democratic candidate who lost in a landslide in a state election in 2022?

   Example Answer: Patricia Mazzei wrote the New York Times article "With Runaway Win, DeSantis's Political Career Becomes Supercharged" published on November, 9th 2022 which mentions Charlie Crist who lost the Florida gubernatorial election by a landslide margin.

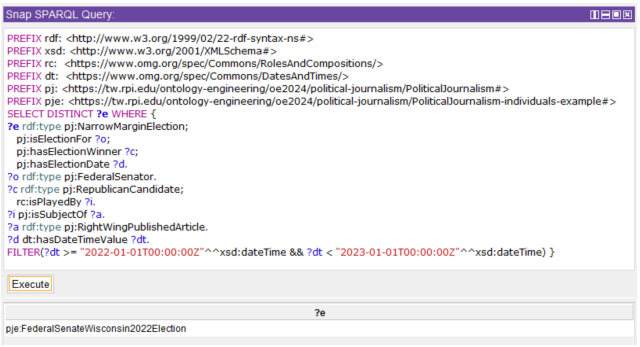
   Reasoning: We will query for Democratic candidates who lost an election where the election was a state election and the margin was a landslide. We will then see if the person who plays the role of that candidate is mentioned in any article by the New York Times. If there is such an article, the relevant data, including the name of the journalist, will be returned.

5. What other topics in articles published in 2024 are covered by journalists who have published articles on immigration and Kamala Harris for right-wing news outlets also in 2024?

   Example Answer: "Harris visits crucial border state as immigration record sparks scrutiny: A timeline written by Adam Shaw" by Adam Shaw mentions both Kamala Harris and the US Southern Border. Adam Shaw also authored "GOP Rep Schweikert projected to fend off Dem challenger in key Arizona House race" which covers the 2024 federal House election in Arizona.

   Reasoning: We will query for journalists who work for right or lean-right publishers and see if they have articles from 2024 which mention both Kamala Harris and an immigration topic. If there is such an article, then the topics which have been written about by the journalist in other articles will be returned, alongside the name of the journalist and the publisher.

An example of a Snap SPARQL query used for testing the first competency question with the examples ontology is shown in the following figure:



5 Discussion

5.1 Value of Semantics

We include semantics to help to simplify user queries and to infer additional information from what is given using encoded Description Logic. This allows users who are less familiar with SPARQL to still ask complex questions. It also allows users to write queries quicker and in a way that looks more similar to natural language. The inference of additional information helps to make explicit what is implied.

While it is declared explicitly that a candidate is running in a specific election and that an election is for a specific office, the idea that a candidate is running for a given office is an important semantic addition to our ontology. To make this inference, we must create a super-property chain. The *isCandidateIn* property relates a candidate with an election they are in, and the *isElectionFor* property relates an election with the political office it is for. If a candidate is running in an election and that election is for a political office, we can conclude that that candidate is running that political office. We can therefore represent the property *isRunningFor* (which relates a candidate to a political office they are running for) as a chain of the isCandidateIn and *isElectionFor* properties. This chain took the form, "*isCandidateIn* o *isElectionFor* → *isRunningFor*". This inferred property allows for users to talk about a candidate running for an office in a simplified way without having to manually chain properties together in SPARQL.

As news outlets are often referred to as left-wing, right-wing, or centrist, we needed a broader approach to media bias than the granularity given by AllSides, which represents bias through five labels: left, lean-left, centrist, lean-right, and right. For that reason, we classified left-wing publishers as any publisher who had a left or lean-left AllSides media bias, right-wing publishers as any publisher who had a right or lean-right AllSides media bias, and centrist publishers as any publisher who had a centrist AllSides media bias. We did this through the construction of three classes of media bias: *LeftWingMediaBias*, *RightWingMediaBias*, and *CentristMediaBias*. The AllSides media bias labels were

created as individuals and assigned to each class according to the aforementioned mapping structure. This then allowed the inference of left-wing, right-wing, or centrist publishers based on if they had some media bias in the created classes. This inference allows for asking more generalized questions without having to specify the individual instances of media bias.

This also allowed for additional semantic inferences with respect to the articles published by the outlets. While we cannot refer to individual articles as having a left, right, or center bias without first analyzing the content of the article, we can state that there are articles that are published by outlets with a specific bias. For this reason, we created inferred classes: *LeftWingPublishedArticle*, *RightWingPublishedArticle*, and *CentristPublishedArticle*. Each of these classes are inferred based on the bias of the article's publisher and allow for the simplification of queries as these are widely useful classes in looking for discrepancies between articles from outlets of differing political persuasions.

Primary and general elections are distinct but related elections for an office. Oftentimes it can be useful to query for one but not the other, as such we have encoded inferred properties more granular than *isElectionFor*: *isPrimaryElectionFor* and *isGeneralElectionFor*. To infer these properties, we used a method known as "rolification" as described in Krisnadhi, Maier, and Hitzler's "OWL and rules" (2011). This is when a class is given an inferred role which relates every member of that class with itself, in our case we created *isPrimary* and *isGeneral*. Using these inferred class properties, we could then link them in a super-property chain with *isElectionFor* to infer *isPrimaryElectionFor* and *isGeneralElectionFor*. These chains took the form of "*isPrimary* o *isElectionFor* → *isPrimaryElectionFor*" and "*isGeneral* o *isElectionFor* → *isGeneralElectionFor*". While it makes intuitive sense for *isPrimaryElectionFor* and *isGeneralElectionFor* to be subclasses of *isElectionFor*, they unfortunately cannot be subclasses thereof as it violates the restrictions imposed on RBox as described in Rudolph's "Foundations of Description Logics" (2011). The inverses of *isPrimaryElectionFor* and *isGeneralElectionFor* were also inferred; they were *hasPrimaryElection* and *hasGeneralElection*. These also cannot be subclasses of *hasElection*, as the inverse of a non-simple role is itself non-simple. These inferences allow for queries that are interested in only one type of election without unnecessarily complicating the SPARQL query.

Seeing as general elections and primary election are connected to each other in determining who occupies a given office for a term, we used inference in order to connect related primary and general elections through the properties: *hasRelatedPrimaryElection* and *hasRelatedGeneralElection*. These are inferred through the assumption that if a general election and a primary election are both elections for a given term of a given political office then they must be related. These inferences used the same rolification method described above and used the super-property chain: "*isPrimaryElection* o *isElectionFor* o *hasElection* o *isGeneralElection* → *hasRelatedGeneralElection*" and "*isGeneralElection* o *isElectionFor* o *hasElection* o *isPrimaryElection* → *hasRelatedPrimaryElection*". These chains look at the given type of election, then see what office (and term) this election was for, then see what other elections are also for this office term, then check if those are of the other election type, if they are then the relationship is inferred. These inferred properties allow the connection of related primary and general elections despite their relationships not being explicitly stated.

Primaries are often used within one political party for determining what candidate they will choose for the general election. For this reason, most primaries have candidates exclusively of one party. This allows us to infer based upon the party of the candidates if a primary is a Republican primary or a Democrat primary. If all of the candidates for a primary election are Republican, then we can infer that it is a Republican primary, and likewise for the Democrats. These inferred classes allow for the query of a specific political party's primary elections.

An additional semantic inference is if a candidate in an election is a Republican candidate or a Democrat candidate. First this required creating classes to represent Republican versus Democrat candidates. This is easily inferred based on the political party of the candidate and the *RepublicanCandidate* class is defined as equivalent to "*PoliticalCandidate* and (*hasPoliticalParty* value *RepublicanParty*)" and similarly for the *DemocratCandidate* class. To then create a property that links Republican or Democrat candidates to an election, we must use rolification again. The rolified properties are *isRepublicanCandidate* and *isDemocratCandidate* and the chain took the form, "*isRepublicanCandidate* o *isCandidateIn* → *isRepublicanCandidateIn*" for the Republican candidates and "*isDemocratCandidate* o *isCandidateIn* → *isDemocratCandidateIn*" for the Democrat candidates. These inferred properties could likewise not be subclasses of *isCandidateIn*. The inverse of these have been added as *hasRepublicanCandidate* and *hasDemocratCandidate*. This allows for simplified queries as well as additional readability when looking at the elections in protégé.

A simple election is an election which has no more than one winner; this is also known as a single-winner election. A won election is an election in which a winner has been determined. We can infer that a given candidate is an election loser if they are not the election winner in a won simple election. We cannot assume that just because an election has only one winner that all other candidates are losers as it may be a multi-winner election where one winner is announced before the rest. If we were to load the ontology at an arbitrary time where that one winner was

already announced but the others were not, we would not want to conclude that others were all election losers. For this reason, we restrict the inference of election losers to exclusively simple elections. For a similar reason, we cannot infer simple elections based on the number of election winners as it may be a multi-winner election where some winners have yet to be announced. We can, however, infer that all Federal US elections are simple elections.

We made additional inferences based upon what office a candidate is running for or what office an election is for and labelled them as federal, state, or local.

5.2 Limitations

The primary limitation of our ontology is that there is no standardized article keyword taxonomy that is adhered to by multiple publishers. This made the process of linking various articles to their mentioned subjects and topics far more difficult than we initially anticipated. Additionally, while our original goal was a system that would shine light on political bias in media, the system only can compare the discrepancies in publishing patterns across various media outlets and does not correlate these patterns with a particular political bias. This puts the onus on the user to craft questions and queries that expose differences in media coverage that might correlate with bias.

Finally, this practical use of this system is heavily reliant on the relevance of the news articles and elections represented. While the web scrapers and API callers we built to ingest the data should continue to work as long as the APIs remain backwards compatible, there is a possibility that the scrapers cease to work as intended after some time due to changes in API responses.

5.3 Website

More information regarding our ontology including terminology, concept model diagrams, and our full use case description can be found at.
https://journalism--rpi-ontology-engineering.netlify.app/oe2024/political-journalism/

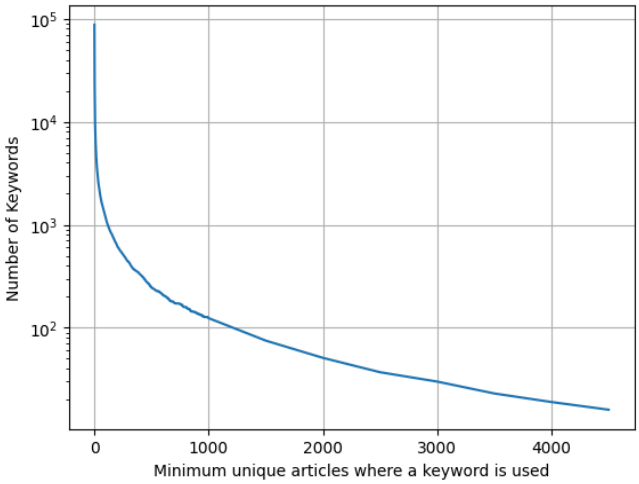
6 Future Work

6.1 Standardized Article Keywords

One of the questions that initially inspired this project was whether media bias can be observed in the coverage patterns for specific topics across different news outlets. A major challenge in answering this question was defining what constitutes a media outlet "covering" a specific person, topic, or event. The problem is that each news outlet has its own process for generating keywords, maintaining an internal taxonomy, and assigning known keywords to describe articles. Another way to conceptualize this problem is that if the same exact article was published by *The New York Times* and *Fox News* it is likely that each publisher would associate a different set of keywords to the article. Ideally, there exists a way to use each news outlet's taxonomy of keywords to bridge the gap in how the same article might be represented. In the following sections we will cover some of the potential solutions to this problem that we hope to continue in future work.

6.2 Keyword Statistics

Before we attempt to standardize article keywords it is useful to quantify the scale of the problem by looking at the unique keywords tagged in NYT articles from 2020 to 2022.



We can see that over 80,000 unique tags are used to describe the articles with over half of these tags being used only once. There are 123 keywords that were used to tag more than 1000 unique articles. These heavily used keywords often reflect overarching topics like "'United States Politics and Government'" or commonly written about people like "'Trump, Donald J."

6.3 Tag Generation from Mining Relative Use

One solution we explored was to extract how often the most commonly used keywords appeared in the same articles. First, we filtered out all keywords that were used to tag less than 100 unique articles. Then, with the remaining 1141 keywords we generate a 1141 x 1141 matrix, N, where entry N_{ij} represents how many articles contain keyword i and do not contain keyword j. We then normalize N_{ij} by the total number of times keyword i appears in any article to get our exclusive use ratio. This exclusive use ratio tells us how often a keyword is used independently of another keyword. The assumption here is that if *keyword_1* almost always appears in articles with

keyword_2 but there exist a relatively larger number of articles where *keyword_2* is used without *keyword_1* then we *might* infer that *keyword_1* has some subclass relationship to *keyword_2*.

We'll use the following example with keywords "TikTok" and "Social Media" to illustrate this idea.

"TikTok" has an exclusive use ratio of .21 with respect to "Social Media." This means 21 percent of articles with "TikTok" as a keyword do not have "Social Media" as a keyword. Additionally, when manually examining the content of articles that contain both keywords and articles that contain only "TikTok" there does not appear to be any obvious difference in the other keywords used or the topic of the article.

By applying an exclusive use ratio of 0.25 to all keywords we automatically generate the following potential subclasses of "Social Media."

```
['Facebook Inc', 'Instagram Inc', 'TikTok (ByteDance)', 'Twitter', 'Zuckerberg, Mark E']
```

We acknowledge that this approach for generating topic and keyword subclasses is not perfect and can be corrupted by concentrated media cycles when two distinct topics happen to be connected by a major news story. For example, this algorithm incorrectly assumes "Impeachment" is a subclass of "Trump, Donald J" because almost all articles involving impeachment over the time span we looked at were referring to the impeachment of Donald Trump. Still this system allowed us to infer 105 top level article classes with over 300 subclasses. Additionally, there is no theoretical reason that this process cannot be applied recursively on less and less common keywords to generate more levels to the taxonomy. The most straightforward solution to the problem of incorrectly generated subclasses is to gather data over a greater time horizon to prevent influence from specific news cycles. Another idea is to use the simple taxonomy provided by the NYT to create rules preventing certain subclass relationships. An example of such a rule would be preventing subclasses of descriptors (e.g. impeachment) from becoming a subclass of any person (e.g. Donald Trump).

6.4 Tag Generation from Keyword Embeddings

Another approach to generating keywords from the articles involved a fusion of word embedding techniques taken from Natural Language Processing and dimensionality reduction techniques. We used Google's Bidirectional Encoder Representations from Transformers (BERT) model to transform each keyword into a 768 dimension vector embedding. While the exact generation of these embeddings from the keyword is a somewhat black box process, many of the dimensions in the embedding space connect to real world concepts like person names and broader topics in government. Our approach was to convert the most common keywords into embedding vectors, scale each of these vectors by the number of unique articles containing that keyword, and then combine these vectors as columns of a large, 768 x 1141 matrix (768 is from the dimension of the embeddings vectors and 1141 is from the number of commonly used keywords). We perform principal component analysis on this scaled embedding matrix to get the principal components (i.e. directions) that capture the most variance in the keywords. Finally, we search for the keyword with the embedding that has the closest cosine similarity to each principal component. This gives us a list of the most likely top level keywords in order of how much they capture the variance in the article keywords. The following are the top level keywords corresponding to the 20 largest principal components.

```
['Politics and Government', 'Coronavirus Aid, Relief, and Economic Security Act (2020)', 'Coronavirus (2019-nCoV)', 'United States Politics and Government', 'Biden, Joseph R Jr', 'Books and Literature', 'Republican Party', 'New York City', 'Demonstrations, Protests and Riots', 'Content Type: Personal Profile', 'Social Media', 'Race and Ethnicity', 'New York State', 'Presidential Election of 2020', 'Immigration and Emigration', 'Economic Conditions and Trends', 'Hygiene and Cleanliness', 'Immigration and Emigration', 'United States Defense and Military Forces',...
```

We were initially concerned that these results were simply the 20 most commonly used keywords, but 7 out of 20 automatically selected keywords do not appear in the list of commonly used keywords. Furthermore, the ordering of the automatically selected keywords is not simply based on the occurrence count. The goal of this approach was that we could generate a basis of keywords for each media outlet's keyword taxonomy and then perform a change of basis to convert articles from one publisher's tags to another. However, this technique quickly loses effectiveness on most proper names, lesser known events, and for distinguishing keywords that have similar words or spellings but refer to very different concepts.

6.5 General Ontology Additions

An important addition is the support for terms of political offices existing over a certain time interval and a person would play that role for that time interval.

An important semantic addition would be to infer an election winner agent probably plays the role of a term of office. The reason the general election winner can only be a probable office holder is due to the threat that the winner may die before taking office or be otherwise unable to take office. Another reason is that the winner may simply decline appointment to the office, although this mainly happens for write-ins at the local level.

Another semantic addition would be to infer a primary

candidate is probably a candidate in the general election. The reason the primary election winner can only be a probable candidate is likewise due to the threat of the candidate being unable to fill the role or declining that role.

7 Conclusion

Ultimately, our attempt to create an ontology for analyzing political news articles involved an enormous number of data sources and unexpected modeling challenges that under the time constraints we had extreme difficulty integrating the various parts of the ontology. While the actual usability of the system remains far below what we initially intended, we hope this work serves as a foundation for exploring the various data sources, techniques, and conceptual models that might be useful in future approaches to build such a system. Additionally, we hope the deficiencies we have pointed out in the current standards for news article syndication shine a spotlight on the potential to add more consistency in article keyword tagging and semantic capabilities across media outlets.

Acknowledgements

This project would not have been possible without the support and guidance from our RPI Ontology Engineering course instructors Professor Deborah McGuinness and Ms. Elisa Kendall as well as the exceptional feedback, mentorship, and patience from project mentors, Jade Franklin and Danielle Villa. Additionally, we thank journalist Timmy Facciola for graciously providing domain expertise on the media publishing process and giving feedback on potential uses for this system. Finally, the data for this project would not have been accessible without the generosity of *The New York Times* for providing free access to the historical archive API and for the Internet Archive's free access to the WayBack Machine API.

References

Bødker, H. 2018. Journalism History and Digital Archives. *Digital Journalism*, 6(9), 1113–1120. doi.org/10.1080/21670811.2018.1516114

Krisnadhi, A., Maier, F., Hitzler, P.: OWL and rules. 2011. *Reasoning Web*. LNCS, vol. 6848: 382–415. doi.org/10.1007/978-3-642-23032-5_7

Moreira, S., Batista, D., Carvalho, P., Couto, F.M., Silva, M.J. 2011. POWER - Politics Ontology for Web Entity Retrieval. In: Salinesi, C., Pastor, O. (eds) Advanced Information Systems Engineering Workshops. CAiSE 2011. Lecture Notes in Business Information Processing, vol 83. Springer, Berlin, Heidelberg. doi.org/10.1007/978-3-642-22056-2_51

Musil, T. 2019. Examining Structure of Word Embeddings with PCA. In: Ekštein, K. (eds) Text, Speech, and Dialogue. TSD 2019. Lecture Notes in Computer Science(), vol 11697. Springer, Cham. doi.org/10.1007/978-3-030-27947-9_18

Rudolph, S.: Foundations of description logics. 2011. *Reasoning Web*. LNCS, vol. 6848: 76–136. doi.org/10.1007/978-3-642-23032-5_2