Machine Coding with the Institutional Grammar

The Institutional Grammar (IG) is used to analyze the syntactic structure of statements constituting institutions (e.g., public policy and social norms) that indicate behavioral constraints. Significant progress has been made in methodologically developing the IG in recent years. Scholars have offered increasingly clear guidelines for IG-based coding, identified unique considerations for applying the IG to different types of institutions, and even expanded its syntactic scope. However, while validated as a robust institutional analysis approach, the resource and time commitment associated with its application has precipitated concerns over whether the IG might ever enjoy widespread use. Needed now in the methodological development of the IG are reliable and accessible (i.e., open source) approaches that reduce the costs associated with its application. We propose an automated approach leveraging computational text analysis and natural language processing. We then present results from an evaluation in the context of food system regulations.

Keywords: institutional analysis, institutional analysis and development framework, natural language processing, computational linguistics, food policy

I. Introduction

In 1995, Sue Crawford and Elinor Ostrom introduced an approach for analyzing the language of institutions that govern behavior (e.g., public policies and social norms), called "A Grammar of Institutions" (Crawford and Ostrom, 1995). A primary motivation for the Grammar of Institutions, referred to hereafter as the Institutional Grammar, was to offer a systematic basis for defining different types of institutions that institutional analysts had been investigating in the context of laboratory (e.g., game theoretic) and field settings; namely, strategies, norms, and rules (North, 1990). Crawford and Ostrom argued that institutions of different types have qualitative differences that can be discerned through examinations of the language through which they are expressed. That is, when strategies, norms, and rules are communicated, either orally or in written form, they convey distinct sets of linguistic elements. Based on this observation, Crawford and Ostrom constructed the Institutional Grammar around a set of syntactic elements common to institutions, with each element conveying a distinct type of behaviorally relevant semantic meaning. *Strategies*, they posit, convey actions, actors associated with those actions, and temporal, spatial, and procedural conditions of actions. *Norms*, convey all of this information, as well as indicate whether actions are required, allowed, or forbidden. *Rules*, convey all of the same information as norms, but also specify incentives for action/inaction. Rooting institutional variation in parameters and incentives of action not only provided a basis for differentiating strategies, norms, and rules, it also offered a nuanced approach for specific assessments of how institutions are intended to affect behavior.

The IG received little scholarly attention immediately following its introduction, but interest in it has burgeoned over the last ten years (see Siddiki et al. (2019) for a meta-review of published

IG research). Scholars from around the world have applied the IG in a variety of institutional settings (Basurto et al., 2010; Feiock et al., 2016; Roditis et al., 2015); demonstrating its utility in differentiating types of institutions, as well as for gaining systematic and comprehensive understanding of institutional design. They have also demonstrated how the IG can be leveraged to operationalize key concepts relating to governance, such as institutional coerciveness and institutional change (Siddiki, 2014; Weible and Carter, 2015). Finally, through various applications, scholars have showcased the theoretical and methodological versatility of the IG. They have shown, for example, that the Institutional Grammar can be effectively paired with various concepts, theories, and methods in the study of governance phenomena. The IG has been used to operationalize concepts and theories relating to polycentricity (Heikkila and Weible, 2018), policy design (Siddiki et al., 2011), policy implementation (Carter et al., 2015), nonprofit management (Siddiki & Lupton, 2016), and a host of other domains of study (Clement et al., 2015). Methodologically, the IG has been coupled with interviews and surveys (Siddiki et al., 2012), as well as used in the context of computational simulations to understand behavioral outcomes linked to policy design (Frantz et al., 2015; Ghorbani and Bravo, 2016). In all, the IG has enjoyed a robust revival in the last decade.

In most of this existing research applying the IG, scholars have used it to study the design, and related impacts, of institutions codified in written form; in particular, institutions taking the form of public policy (Basurto et al., 2010; Carter et al., 2016; Feiock et al., 2016). This line of research was initiated by Basurto et al. (2010), who used the IG to study the design of abortion and transportation policies in the U.S. state of Georgia. Importantly, in addition to providing the first empirical application of the IG to study public policies, Basurto et al. presented

the first set of coding guidelines for utilizing the IG for this purpose. Others have added to their effort by contributing additional coding guidance (Siddiki et al., 2011; Carter et al., 2015).

The coding guidance offered by Basurto et al. and others was critical for the methodological development of the IG. This coding guidance laid out clear practical steps for applying the IG, and in so doing, contributed to greater validity and reliability in IG-based coding of formal institutions, such as public policies. Yet other methodological challenges associated with applying the IG remain that have yet to be fully resolved. Among the steepest is the significant capital investment associated with applying it. With the exception of Heikkila and Weible (2018), all published applications of the IG involving the study of public policy rely on fully manual (i.e., human) coding of policy text. Manually parsing public policies into individual directives (referred to as institutional statements in the IG parlance), and then further classifying the language of these directives along the IG's syntactic elements requires a significant amount of time and resources. The implication of this is two-fold. First, it limits the amount of data that analysts can collect. Most studies using the IG report analyses of a single policy or a limited number of policies, which inherently raises concerns about the generalizability of findings emerging therefrom. A second implication is that it inhibits the widespread uptake of the IG; one confronts a steep learning curve to understand coding the IG, then must actualize that in parsing individual documents passage by passage.

One way to overcome the resource constraints of applying the IG is to automate the coding process. Machine-coding policy texts can vastly reduce the amount of human effort and time commitment associated with applying the IG, thus enabling the study of large quantities of public policies from which generalizable insights relating to policy design can be derived. Recent

advancements in computational linguistics, and related natural language processing and machine learning techniques, can be effectively leveraged in the effort to automate the IG coding process. In this paper, we report evidence of the effectiveness of this approach, and present the findings from a pilot study focused on developing an automated approach for coding public policies based on the IG using natural language processing. Findings from the pilot study show that: (i) automated coding achieves coding accuracy values on individual IG syntactic elements that are comparable to those achieved through manual coding; (ii) automated coding can be effectively applied to policies of various types (e.g., statutes, regulations, executive orders) and with variable foci and structural characteristics; and (iii) preprocessing of original policy texts improves accuracy of automated coding.

This paper proceeds as follows. Part 2 provides an elaborated description of the IG and its main elements. This description is followed in Part 3 by an overview of applications of the IG. This overview highlights the challenges associated with manual applications of the IG. Part 4 introduces natural language processing and automated coding. Part 5 describes how we prepare policy texts for analysis. Part 6 describes how we leverage NLP techniques to build an automated IG classifier. In Part 7, we present and discuss the results of our pilot study on food system regulations. Part 8 concludes the paper.

II. Introduction to the Institutional Grammar

The focal unit of analysis under the IG is the institutional statement, which is parsed along six syntactic elements. According to Crawford and Ostrom (1995, 583):

> The term institutional statement refers to a shared linguistic constraint or opportunity that prescribes, permits, or advises actions or outcomes for actors (both individual and corporate). Institutional statements are spoken, written, or tacitly understood in a form intelligible to actors in a corporate setting.

Within the context of written institutions, like public policies, institutional statements are often coterminous with sentences, but not always. If a sentence addresses multiple actors or actions, it may contain multiple institutional statements, which is defined based on the presence of specific syntactic elements, as explained in more detail below. The following is an example of an institutional statement: "Operations certified as organic under the U.S. National Organic Program must submit an organic system plan annually or face certification revocation."

The six syntactic elements comprising the IG, along which statements are parsed, include the following: (1) *Attribute* [A], the actor to whom an institutional statement applies; (2) *Aim* [I], the action of the statement; (3) *Deontic* [D], the prescriptive operator that indicates whether the Attribute is required, forbidden, or permitted to perform the focal action of the statement; (4) *Object*[1] [B], the animate or inanimate receiver of the focal action; (5) *Condition* [C], the temporal, spatial or procedural boundary (or boundaries) in which the action of statement may, must, or must not be performed; and (6) *Or else* [O], incentives for performing/failing to perform the focal action (e.g., the punitive sanction associated with not carrying out the focal action as prescribed). Syntactic elements are sometimes summarized with the acronym "ABDICO." The following shows how the example institutional statement offered above—"Operations certified as organic under the U.S. National Organic Program must submit an organic system plan annually or face

[1] The Object was introduced by Siddiki et al. (2011).

certification revocation"—would be deconstructed along Institutional Grammar syntactic elements: *Attribute* = "operations certified as organic under the U.S. National Organic Program"; *Deontic* = "must"; *Aim* = "submit"; *Object* = "organic systems plan"; *Condition* = "annually"; *Or else* = "or face certification revocation."

At a minimum, institutional statements contain an Attribute, Aim, and Condition. Consistent with their objective of offering a way for differentiating between types of institutional statements based on linguistic elements—rather, the presence or absence of different elements—Crawford and Ostrom characterize an institutional statement containing an Aim, Attribute, and Condition as a *Strategy*, an institutional statement containing an Aim, Attribute, Condition, and Deontic as a *Norm*, and an institutional statement containing all syntactic elements as a *Rule*. An Object may be present in any type of statement. Deontics and Or else clauses are the key elements for distinguishing rules from norms from strategies.

III. Applications and Challenges in Manually Coding the IG

Though Crawford and Ostrom introduced the IG in 1995, it wasn't until 2008 that the first journal article that addressed it was published. Between 2008 and 2019, twenty-one journal articles have been published that address the IG. Substantively, extant research can be grouped according to four, sometimes overlapping, thematic foci. One set of articles is focused on the behavioral assumptions and ontological logic underlying the IG syntax (e.g., Schlüter & Theesfeld, 2010). Another set of articles focuses on leveraging the IG within the context of computational simulation; specifically, within agent-based models, with the objective of assessing endogenous emergence of institutions (e.g., Frantz et al., 2015; Ghorbani & Bravo,

2016). In these studies, the IG is used in the parametrization of models. The IG interfaces naturally with agent-based modeling since many of the syntactic elements correspond to key modeling parameters. For example, agent and agent roles are reflected in Attribute data. Agent activities are captured in Aims, and the Conditions specify the parameters that define when/where/how these activities occur. A third set of articles focus on the IG as a methodological tool and offer suggestions for its methodological refinement (Basurto et al., 2010; Carter et al., 2015; Siddiki et al., 2011). These articles also often contain "proof of concept" empirical applications. Finally, but most relevant to the exercise reported in this paper, a fourth set of articles focuses on applying the IG to study the design of policies, such as laws, regulations, and city charters, and implications thereof (e.g., Basurto et al., 2010; Siddiki, 2014; Feiock et al., 2014).

In studies that apply the IG to evaluate policy design, authors typically dissect policy texts manually into institutional statements and then further along IG syntactic categories. These hand-coded data are then analyzed in accordance with the authors' research objectives. Authors of such studies often report on the laborious and time-consuming nature of manual coding (Siddiki et al., 2019). As noted previously, this reality limits the amount of data that can be collected, and can also dissuade scholars from embarking on research that employs the IG.

One resolution to the posited problem is to automate the coding process; that is, rely on computers to code policy texts in line with the IG syntax. One of the first published efforts in this area used a semi-automated approach that adapted and applied the IG to analyze thousands of institutional statements and categorize them by institutional types for describing polycentricity (Heikkila & Weible, 2018). Recent advances in machine learning and natural language

processing platforms offer great promise in facilitating automated coding. We turn to an overview of these now.

IV. Natural Language Processing (NLP) and Automated Coding

NLP is a subfield of artificial intelligence (AI) and linguistics that engages computers to understand human language. It primarily utilizes two groups of techniques to interpret human language data: syntactic and semantic analysis. Syntactic analysis pertains to evaluating natural language in relation to structural elements and grammatical rules. Key analytical tasks associated with syntactic analysis include tokenization, lemmatization, stemming, and part-of-speech tagging. Tokenization involves splitting a large set of texts into tokens, a sequence of characters that are grouped together in a useful way—often individual words. Both lemmatization and stemming reduce various inflected forms of a word into a single form. For example, a group of inflected words “am, is, are” will be replaced by “be”. Part-of-speech tagging identifies the part-of-speech for every token.

Semantic analysis is focused on understanding context-relevant meaning and interpretation of words. A number of more or less guided techniques can be engaged in semantic analysis. A more guided technique is Named Entity Recognition (NER), which is characteristically used to identify names of people, organizations, and places from text. A less guided approach is to employ machine learning techniques to inductively derive semantic meaning. Through this approach, the meaning of particular words is derived based on algorithmically “learned” semantic patterns among surrounding words, as well as syntactic dependencies between words.

To overcome the challenges associated with manual coding of policy documents in accordance with the IG, we employ a variety of NLP techniques in conjunction with methods from machine learning to automatically code policies in terms of the IG. As an overview, Figure 1 outlines the general process of automated coding relating to our exercise. First, a set of raw texts (e.g., policy documents) are "preprocessed." The preprocessed data are then annotated using the NLP program. The program then maps the annotations to IG syntactic elements, based on a supervised learning program trained using a large amount of human-coded data employed in prior published work. After learning is complete, the program filters institutional statements. The resulting output is a simple, standard csv ("comma separated value") format data file that maps institutional statements to their elements in the IG. In the remainder of the paper, we detail each of these steps, then provide some initial evidence of validity as well as discussions of future directions.

Insert Figure 1 about here

V. Preprocessing

In general, text must undergo preprocessing so that it is suitable for downstream analyses. While standard text pre-processing often focuses on elements such as stop word removal or removing punctuation, here we are particularly concerned with issues related to the formatting of policy documents. Specifically, for purposes of Grammar-style analyses, many policy documents present challenges as they consist of sentence styles that differ from the typical prose utilized in many other written contexts; frequently, policy documents contain, for example, various types of outline headers, fragmented clauses, and uncommon punctuation.

Thus, the first step is a preprocessing approach suitable for an automated IG application. During the preprocessing stage, we remove any special characters or text formatting styles that may be problematic for automated coding. We also restructure certain text structures to facilitate downstream NLP tasks. Consider a long, multi-clause statement structured as a bulleted list. Such statements are often difficult for NLP programs to interpret, as their clauses are long and complex, and separated syntactically by non-syntactic (even non-linguistic) document features such as special characters (e.g. bullets) and line-breaks. The clauses of a bulleted statement can serve very different functions: establishing context for subsequent clauses, detailing steps of a sequence, aspects of a requirement, or standing independently as self-contained statements. Failure to recognize these functions during preprocesses will set later stages—namely, dependency parsing—up for failure. Being relatively common in policy documents, and rare in the types of documents that standard NLP tools were designed for, these features therefore require special handling, a central task of the preprocessing stage

Figure 2 demonstrates the general workflow involved in preprocessing. It is notable that this does not include steps necessary for the user to undertake contextually before any application of automated methods. Preprocessing typically engages a generalizable search function, a specific code that is based on a set of user-specified rules to identify and remove headings, outline fragments (e.g., roman numerals), and irregular punctuation. This generalizable, rules-based search function is called a regular expression. Document headers, outlined subheadings, page numbers, and signature lines should be manually deleted. After a text has undergone preprocessing, it is exported as JavaScript Object Notation (JSON) files, a data format for flexibly representing complex datasets in terms of nested or tree-like structures, whose

complexity represents an intermediate stage toward the tabular csv format necessary for statistical analysis.

Insert Figure 2 about here

VI. Automating Institutional Grammar Coding

After preprocessing is done, the next step is to classify the restructured policy text into the IG. To do so, we rely on the general insight that the elements of the IG —as well as other general textual characteristics—map to standard grammatical constructions. That being the case, we rely on the well-developed literature around standard NLP approaches to annotate texts, then leverage those annotations to predict the IG elements associated with each word in the policy text. Our approach more specifically involves three steps: computing NLP annotations, matching them to the IG's syntactic elements, and extracting institutional statements. The basic intuition is to link the NLP program's analysis information to IG syntactic elements and filter out institutional statements. In this section, we begin by describing the annotation of the documents, then describe two approaches we have taken to mapping the annotations to the IG and the comparative success of those approaches at accurately predicting the human-assigned IG elements.

We start with annotation. To do so, we rely on Stanford's CoreNLP (Manning et al., 2014), a well-developed suite of NLP tools for identifying a broad range of different characteristics in texts. CoreNLP is the gold standard for automated grammatical analysis and is open-source software. As such, it is perfectly suited for our task.

We implement CoreNLP through the R statistical programming language. The capability of R for NLP and other text-as-data tasks has expanded significantly, and continues to do so. During the annotation step, the preprocessed texts run through CoreNLP's annotation pipeline, including sentence splitting, Part-of-speech tagging, Named Entity Recognition, and Syntactic Parsing. For each word in the document, we therefore gain a host of potentially relevant information about the word itself, and its relation to nearby words.

The second step is taking these characteristics and associating them with the IG. We do this by probabilistically associating syntactic elements with IG elements using supervised machine learning. To motivate this complex approach, consider first a simpler approach: treating syntactic characteristics as *uniquely* predictive of IG elements, such that a particular dependency, or part-of-speech, or other characteristic, predicts one and only one grammatical component. This approach relies on a predefined mapping of the characteristics identified by CoreNLP to the IG elements. To characterize this approach, we ran a dependency parser to deconstruct sentences and define their grammatical structures and the relationships between their words. Table 1 provides a hypothetical version of the mapping of the assigned dependencies to the IG ABDICO elements.

This sort of one-to-one mapping has the advantage of simplicity, but at the cost of lost information and the inability to handle any higher dimensionality problems. To understand the stark limitations, consider a hypothetical word that appears in two different contexts within a document and is associated with two different IG codings within that same document; identifying a one-to-one match of the word to the IG code is therefore impossible. We might rely instead of the identified relations, but again we expect that relations will only be correlated with

rather than perfectly predictive of a particular component classification. We could, of course, develop such a mapping ourselves, writing a series of decision rules. The costs of doing so are exorbitant however, and as a result quite limited.

Insert Table 1 about here

We turned, therefore, to the supervised learning approach. Supervised learning approaches operate by training a classifier that links the identified characteristics from the NLP analysis to the human-coded IG elements, in order to try to mimic human discriminations. We begin by partitioning the data into testing and training sets. The training set is the dataset on which we "train" the classifier to predict the human-coded IG, using all of the features of the text as predictors in the algorithm. Then, we assess the accuracy of the trained classifier using the held-out test set.

In all, our supervised learning approach operates as follows. First, we utilize Stanford CoreNLP's dependency parser and Part-of-speech tagger in order to create (for each word in the corpus) a set of features.[2] We convert the classifications into a series of indicator variables. Because CoreNLP assigned each word to at least one of these dependencies by the parser, we are able to create a series of indicators for each of those that simply indicate whether or not the observed word was assigned that dependency. We also include indicator variables for each unique word in the corpus.

[2] For implementing CoreNLP, we rely on the cleanNLP package in R (Arnold, 2017).

Our analysis retains every word with a human-assigned IG label. We then randomly split the word into training and test sets, with approximately 90 percent of the data (in the analyses below, that leads to N = 8,320 classified words) in the training set and 10 percent of the data (again, in the analyses below, that leads to N = 922) reserved for evaluation of the fitted model. With the training set, we train our model to predict the word-Grammar classifications.

Of the many potential approaches for training a supervised learning model, we rely on neural networks.[3] The underlying intuition of supervised learning is that we know a host of characteristics about each individual classified word; our goal then is to infer rules from the classified data that best map the characteristics to the assigned classification. In neural networks, the idea is to model the decision-making process of making a classification in a (very) simplified version of the way human brains work; we create multiple layers of "neurons" (called units below) that communicate while trying to accomplish some task (in supervised learning, usually classification or prediction). The program learns the connections that best predict the classification, and can amplify those connections while minimizing poorly performing connections. In this way, the approach neatly mimics the intuition of the human based approach noted above, but does so at a scale that would be impossible for humans.

[3] We have also experimented with Support Vector Machines (SVM) classifiers (Vapnik & Lerner, 1963; Boser et al., 1992) — which represent data in a multidimensional space and search for the hyperplane(s) that best separate the data according to the classification data — and Random Forest (RF) classifiers (Breiman 2001) — which rely on ensembling and randomization of many individual decision tree classifiers. In the former case, though SVMs are generally good in high dimensional classification settings (see, e.g., D'Orazio et al., 2014; Caruana & Niculescu-Mizil, 2006) as the size of the training data increased, SVMs performed slightly worse than neural networks on out-of-sample prediction tasks, suggesting they would not scale well as we expand our training data in future work. In the latter case, RF models performed slightly worse while taking many orders of magnitude longer to estimate; indeed, some RF models with large training sets were taking over 6 hours to estimate. Again, then, they did not seem likely to scale well. Finally, we explored linear and non-linear ensembles of SVMs and RFs, along with other approaches; none outperformed the neural network approach, while all combinatorials of different approaches magnified computational time.

To train our neural network classifier, we employ Keras (Chollet et al., 2015) and TensorFlow (Abadi et al., 2015), both of which are open-source and are available to be run through R. We train a sequential model with a linear stack of three layers; the first is a dense layer of 256 units, and the second a dense layer of 128 units. For each, we rely on rectified linear unit activation. The third layer is a dense layer of 6 units, equal to the number of classes that we are trying to predict. For this final layer, and only this final layer, we rely on a softmax activation function.The model was trained using Adam (Kingma and Ba, 2017) and a sparse categorical cross-entropy loss function.[4]

The model was trained for 10 epochs. In Figure 3, we plot the loss (top panel) and accuracy (bottom panel) across the epochs for the 90% of data the model is trained on and a held-out test set of 10%. We pause to note that this is an additional split of the training data only; we retain in reserve the10% of data we previously held out as a "hidden" or "invisible" test set. As the plot makes clear, the neural network continues to learn additional features to aid in prediction over the epochs, though the rate of learning decreases markedly and rather suddenly after only one or two epochs. Yet across this range, there is no change in the accuracy within the validation set. That is, the model is achieving in-sample success by virtue of fitting additional noise; indeed, within sample the neural network is achieving classification accuracies that are at or exceeding 90 percent.. To avoid overfitting, with a particular eye on the hidden test set, we do not train beyond 10 epochs. As an additional mechanism to avoid overfitting problems, we employ a dropout rate of 0.1 between each layer, and a batch size of 64.

[4] All code necessary to replicate these analyses will be posted to the Author's dataverse upon publication..

Insert Figure 3 about here

In addition to evaluating the predictive capacity of the trained neural network, we also leverage the predictions in what might be termed a "meta-model." Here, we leverage the predicted classification probabilities from the neural network in a second stage classifier. Specifically, we add the classification probabilities for each of the six categories to our training dataset, and estimate a gradient boosting (Friedman et al., 2000; Friedman 2001) model using extreme gradient tree boosting, also known as XGBoost (Chen and Guestrin, 2016); using XGBoost in combination with neural networks has been regularly demonstrated to yield state-of-the art predictive accuracy.

The underlying motivation for the second stage is two-fold. First, and pragmatically, the first stage might have a tendency to assign a high overall probability to the most commonly assigned IG component. This maximizes the predictive accuracy at this stage, but has the downside of ignoring the small increases in probability associated with particular sub-categories of components that may be substantively meaningful in classification. The second stage, however, can incorporate that additional signal and results in a substantial increase in predictive accuracy. Second, and more theoretically valuable as we consider avenues for development, we envision this stage as later incorporating the probabilities for context terms; that is, those terms which immediately follow or precede the word we are attempting to classify. In doing so, the classifier will naturally incorporate a type of smoothing that will yet further increase both its accuracy and its utility for practical applications.[5]

[5] We train an XGBoost model using the xgboost package in R. We train a model with a softmax objective to be evaluated by a multiclass log-loss function, where the max depth of the trees is 20, we complete 30

VII. Automated Classification of Food System Regulations

With the classifiers — a neural network approach and a tree boosting approach — in place, we turn to evaluating their predictive fit. Our dataset comes from Siddiki (2014) and is comprised of 19 food policy documents, each of which has been classified according to the IG. Our feature set includes 7,848 total unique features. For training and testing, we randomly split 8,320 words into the training set and the remaining 922 classified words into the test set, forming approximately a 90 percent / 10 percent training / testing split. For each token to be classified, the 7,848 unique features includes indicators for the actual word (under the assumption that some words are likely to be associated with some categories), part-of-speech indicators (under the assumption that particular types of speech like adverbs might be associated with some categories), and relations and source indicators from the dependency parser detailed above. The set is expanded then by incorporating these indicators for the token immediately preceding and the token immediately succeeding the token to be classified; that is, we incorporate the immediate context of each word. Neural networks and deep learning approaches more generally are particularly useful in the context of learning the interdepencies and conditionality between features. The vast feature set therefore provides an avenue to understand how, when a particular word is used in a particular way within a particular context, the probability it belongs to a particular category is greater or lesser.

With this data in hand, we move to assessing the success of our automated procedure at predicting human-assigned IG classifications. We start with the neural network approach. In

---

training passes on the dataset, the step size for each boosting step is 0.1, and a regularization parameter on the weight of 0.08.

Table 2, we include the overall accuracy, precision, and recall of the classifier when we use it with the held-out test data. Start first with the overall accuracy. Of the 922 words in the test set, our preliminary approach correctly classifies 74 percent. Given that general levels of acceptable intercoder reliability range from 70 to 90 percent (see, e.g., Quinn et al., 2010), this accuracy offers strong initial evidence of the potential of our approach. Particularly given the relatively small number of classified words on which this is trained, the associated complexity of the classification task, and the number of extensions to our approach which are still available, we believe the classification accuracy points towards the viability of automated classification of institutional documents into the Grammar.

We pause to highlight two additional important considerations in evaluating the success of the classifier. First, though we treat the human-coded data as a gold standard for evaluation, it is not necessarily the case that each individual classification is precisely correct. That is, were one to train two coders in the IG and assign them the same document, achieving agreement on approximately 75 percent of all classified terms would be deemed largely successful. Second, and really as a corollary of the first, where classifications fail is largely around boundary words like "the" or "a" that are not clearly within one or another category within the Grammar. Removing those terms would significantly increase the accuracy of the classifier.

Insert Table 2 Here

Turning to the results within subgroups, Table 6 also includes information on the *precision* (intuitively, if we classify something as, say, an Aim, how likely is it to be an Aim?)

and *recall* (intuitively, of all the instances of Aim, how many do we correctly identify as Aims?). Importantly, the approach performs well across both metrics and across all subgroups. The model underclassifies Aims, both of which appear far fewer times in coded documents, but is more precise when identifying those classifications. On the other hand, the model overclassifies Conditions and Objects, both of which are the predominant classifications, leading to lower precision but higher recall. In all, though, the only area where the model truly struggles is with the Or / Else category, which is rare and comprises only three actual classifications in the test data.

We turn next to the XGBoost approach. Recall that this approach is simply a second stage that aims to supplement the first stage classification by adding the estimated probabilities from the neural network model to the feature set, and again predicting the classifications. Therefore, our feature set is the same, except it now includes an additional six features which are equal to the estimated probability of class membership for each of the six categories from the trained neural network model. The results of the XGBoost model appear in Table 3.

Insert Table 3 about here.

Interestingly, the boosting approach yields slightly *lower* accuracy overall. As the precision and recall metrics make clear, this appears to be partially attributable to struggles with the classification of Attributes, as those are now correctly classified more frequently as Attributes when the model predicts Attribute (i.e., precision) but far fewer instances are actually identified (i.e., recall). The lack of clear improvement in the boosting model, though

disappointing, also points to the initial success of the neural network at correctly classifying variation in the data and the signals available at this point in time in the training data. Taken together, our results validate the utility of automated approaches for the classification of institutional rules and policy statements into the institutional grammar.

VIII. Conclusion

Proposed more than two decades ago, the IG offered enormous promise for a central subject of study for political scientists and others interested in the systematic study of public policy. However, the exorbitant resource costs necessary for implementing the IG in research settings inhibited its growth as an avenue for scholarly inquiry. Recently, scholars have begun making headway in applying the IG, but the considerable promise of the tool remains unrealized. In this paper, we propose and provide initial evidence for the utility of an automated approach to classifying institutional statements in accordance with the IG. As we show, one can approach high levels of accuracy relatively quickly by employing the growing amount of texts classified into the IG in conjunction with robust tools for natural language processing and improved methods for supervised machine learning.

In all, the results provide strong initial evidence of the potential of supervised machine learning for the automated classification of institutional rules into the institutional grammar. This is particularly so given the following two routes for further improvement. First, expanding the training set to incorporate more of coded policy documents is likely to lead to significant increases in the out-of-sample performance with the test data. Though some simple classification tasks can achieve high rates of accuracy with little training data, more complex classification

tasks require large training data sets. For instance, recent work in sentiment analysis, arguably a more simple classification task, regularly utilizes datasets with hundreds of thousands of training instances (see, e.g., Maas et al., 2011). Though we will never approach that level of data supply, the success of our neural network approach with even a relatively small training set provides important evidence that — as the training data is supplemented by additional work — the approach will move towards being a robust and open solution to a problem that has paralyzed and important avenue of research.

Second, and more directed at our particular specification choices, our present approach samples words randomly; however, a better training approach would sample at the statement level from the coded data. By doing so, one can incorporate the labels of context words in the XGBoost model and that additional predictive information is also likely to yield a significant improvement in our out-of-sample performance in the training data. We plan to incorporate this approach as we continue to build this work towards public release.

References

Abadi, M., and others. *TensorFlow: Large-scale machine learning on heterogeneous systems*, 2015. Software available from tensorflow.org.

Arnold, T. (2017). A Tidy Data Model for Natural Language Processing using cleanNLP. *The R Journal*, *9*(2), 248-267

Boser, B., Guyon, I., & Vapnik, V. (1992). A Training Algorithm for Optimal Margin Classifiers. *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*

Basurto, X., Kingsley, G., McQueen, K., Smith, M., & Weible, C. (2010). A Systematic Approach to Institutional Analysis: Applying Crawford and Ostrom's Grammar. *Political Research Quarterly* 63(3): 523.53.

Carter, D., Weible, C.M., Siddiki, S., & Basurto, X. (2016). Integrating course concepts from the institutional analysis and development framework for the systematic analysis of policy designs: An illustration from the US National Organic Program regulation. *Journal of Theoretical Politics 28* (1): 159-185.

Carter, D., Weible, C. M., Siddiki, S., Chonaiew, S. M., & Brett, J. (2015). Assessing Policy Divergence: How to Investigate the Differences between a Law and a Corresponding Regulation. *Public Administration*, *93*(1), 159-176.

Caruana, R., & Niculescu-Mizil, A. (2006). An Empirical Comparison of Supervised Learning Algorithms. *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*.

Chen, T. & Guestrin, C. "XGBoost: A Scalable Tree Boosting System." In *22nd SIGKDD Conference on Knowledge Discovery and Data Mining*.

Chollet, F. and others. 2015. *Keras*. https://keras.io.

Clement, S., Moore, S. A., & Lockwood, M. 2015. Authority, responsibility and process in Australian biodiversity Policy. *Environmental and Planning Law Journal*, *32*(2), 93-114.

Crawford, S.E.S., & Ostrom, E. (1995). A Grammar of Institutions. *American Political Science Review, 89*(3): 582-600.

D'Orazio, V., Landis, S., Palmer, G., & Schrodt, P. (2014). Separating the Wheat from the Chaff: Applications of Automated Document Classification Using Support Vector Machines.

Political Analysis, 22(2): 224-242.

Feiock, R.C., Weible, C.M., Carter, D.P., Curley, C., DeSlatte, A., & Heikkila, T. (2016). Capturing Structural and Functional Diversity Through Institutional Analysis. *Urban Affairs Review, 52*(1): 129-150.

Frantz, C.K., Purvis, M.K., Savarimuthu, B.T.R., & Nowostawski, M. (2015). Modeling Dynamic Normative Understanding in Agent Societies. *Scalable Computing: Practice and Experience, 16*(4): 355-378.

Friedman, J. 2001. "Greedy Function Approximation: A Gradient Boosting Machine." *Annals of Statistics* 1189-1232.

Friedman, J., Hastie, T., Tibshirani, R. and others. 2000. "Additive Logistic Regression: A Statistical View of Boosting." *Annals of Statistics* 28(2): 337-407.

Ghorbani, A., & Bravo, G. (2016). Managing the Commons: A Simple Model of the Emergence of Institutions through Collective Action. *International Journal of the Commons, 10*(1): 200-219.

Heikkila, T., & Weible, C.M. (2018). A semiautomated approach to analyzing polycentricity. *Environmental Policy and Governance*, *28*(4), pp.308-318.

Maas, A., Daly, R., Pham, P., Huang, D., Ng, A., & Potts, C. 2011. "Learning Word Vectors for Sentiment Analysis." In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies,* pp 142-150.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., and McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 55-60.

North, D.C. 1990. *Institutions, Institutional Change, and Economic Performance*. Cambridge, UK: Cambridge University Press.

Quinn, K., Monroe, B., Colaresi, M., Crespin, M., & Radev, D. (2010). "How to Analyze Political Attention with Minimal Assumptions and Costs." *American Journal of Political Science 54*(1): 209-228.

Rice, D. R. and Zorn, C. (2019). Corpus-Based Dictionaries for Sentiment Analysis of Specialized Vocabularies. *Political Science Research and Methods*, 1-16.

Roditis, M. L., Wang, D., Glantz, S., & Fallin, A. (2015). Evaluating California Campus Tobacco Policies Using the American College Health Association Guidelines and the Institutional Grammar Tool. *Journal of American College Health, 63*(1): 57-67.

Schlüter, A., & Theesfeld, I. (2010). The Grammar of Institutions: The Challenge of Distinguishing Between Strategies, Norms, and Rules. *Rationality and Society*, *22*(4):445-475.

Siddiki, S. (2014). Assessing Policy Design and Interpretation: An Institutions-Based Analysis in the Context of Aquaculture in Florida and Virginia, United States. *Review of Policy Research 31* (4): 281-303.

Siddiki, S., & Lupton, S. (2016). Assessing Nonprofit Rule Interpretation and Compliance. *Nonprofit and Voluntary Sector Quarterly*, 45(4): 156S-174S

Siddiki, S., Weible, C. M., Basurto, X., & Calanni, J. (2011). Dissecting Policy Designs: An Application of the Institutional Grammar Tool. *Policy Studies Journal, 39*(1): 79-103.

Siddiki, S., Heikkila, T., Weible, C. M., Pacheco-Vega, R., Carter D., Curley, C., DeSlatte, A. & Bennett, A. (Forthcoming). Institutional Analysis with the Institutional Grammar. *Policy Studies Journal*.

Siddiki, S., Basurto, X., & Weible, C. M. (2012). Using the Institutional Grammar Tool to Understand Regulatory Compliance: The Case of Colorado Aquaculture. *Regulation & Governance 6* (2): 167-188.

Vapnik, V., & Lerner, A. (1963). Pattern recognition using generalized portrait method. *Automation and Remote Contro*l, *24*, 774–780.

Weible, C. M. & Carter, D. (2015). The Composition of Policy Change: Comparing Colorado's 1997 and 2006 Smoking Bans. *Policy Sciences, 48*, 207-231.

TABLES

Table 1. Matching IG ABDICO elements with Universal Dependencies

ABDICO Component	Universal Dependencies
Attributes	nominal subject (nsubj), passive nominal subject (nsubjpass), clausal subject (csubj),
Object	direct object (dobj), dependent (dep)
Deontic	auxiliary (aux)
Aim	root, passive auxiliary (auxpass), negation modifier (neg), phrasal verb particle (compound:prt), indirect object (iobj), copula (cop)
Condition	nominal modifier (nmod), adverbial modifier (advmod), open clausal complement (xcomp), clausal complement (ccomp), adverbial clause modifier (advcl)
Or else	N/A

Table 2: Out-of-sample Performance of Neural Network Classifier for Predicting Institutional Grammar Components from Nineteen Policy Documents.

Level	*Accuracy*	*Precision*	*Recall*	*N*
Overall	0.74	--	--	922
Aim	--	0.94	0.72	69
Attribute	--	0.75	0.61	80
Condition	--	0.68	0.72	322
Deontic	--	0.91	0.94	32
Object	--	0.75	0.76	416
Or/Else	--	0.33	0.33	3

Note: Results based on classification of held-out test set using neural network classifier trained on a set of 8,120 randomly sampled words.

Table 3: Out-of-sample Performance of XGBoost Classifier for Predicting Institutional Grammar Components from Nineteen Policy Documents.

Level	*Accuracy*	*Precision*	*Recall*	*N*
Overall	0.73	--	--	922
Aim	--	0.95	0.77	69
Attribute	--	0.76	0.51	80
Condition	--	0.66	0.72	322
Deontic	--	0.89	0.97	32
Object	--	0.74	0.76	416
Or/Else	--	0.33	0.33	3

Note: Results based on classification of held-out test set using XGBoost classifier trained on a set of 8,120 randomly sampled words and probability scores from the neural network classifier.

FIGURE LEGENDS

Figure 1. Automated Coding Process. This figure provides the sequence of steps for automating the process of getting unstructured institutional statement coded according to the IG.

Figure 2: Preprocessing Workflow. The first step in preprocessing requires the user to read the text file into the preprocessing interface. Second, by applying a rules-based script based on regular expressions, the user can remove all bulleted symbols (or other punctuation) that might hinder the performance of the NLP program when it is used in later stages. Third, one applies an NLP annotator on the cleaned text. This annotator assigns each word or punctuation mark a unique identifier (a "token" ID), and then, after finding the sentence breaks, assigns each grouping of words a sentence ID. Fourth, the sentence ID is used to reassemble the sentences in the proper order. It is at this stage that the user has the option of adding additional contextual information to each sentence, such as statute location or statement type. Finally, the sentence data is combined in a single file formatted to promote better NLP.

Figure 3: Evaluation of Trained Three Layer Neural Network Model of Institutional Grammar. This figure plots the loss metrics (top panel) and accuracy (bottom panel) for the trained models both within the training sample (blue) and within a held-out evaluation set (green) across the 10 training epochs.