Creating an Automated Event Data System for Arabic Text

Andrew Halterman[*] Jill Irvine[†] Christan Grant[‡] Khaled Jabr[§] Yan Liang[¶]

Prepared for presentation at the ISA Annual Meeting 2018 in San Francisco

Abstract

This paper describes the tools and process for building a new machine-coded event data set of Arabic news text. Data produced from text is becoming one of the most important new sources of information for quantitative political science, but most publicly available event datasets are limited to English language sources. The paper describes our new dataset, the process of producing Arabic event data using open source tools and a team of coders, and how researchers can make use of the data. This work stems from an ongoing NSF RIDIR project on "Modernizing Political Event Data," which aims to produce multilingual event data and the software needed for researchers to produce custom datasets.

Introduction

In the past 15 years, automated methods of text analysis have become prominent in political science, allowing researchers to study much larger amounts of text and to find meaning that was difficult to extract manually.[1] Many of these methods analyze documents as a whole, pro-

[*]PhD Candidate in Political Science, Massachusetts Institute of Technology
[†]Professor of International and Area Studies, University of Oklahoma
[‡]Assistant Professor of Computer Science, University of Oklahoma
[§]MA Candidate in Computer Science, University of Oklahoma
[¶]PhD Student in Computer Science, University of Oklahoma

[1]We gratefully acknowledge the support of the National Science Foundation under award number SBE-SMA-1539302, "Modernizing Political Event Data". Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

ducing document-level scores or topics (Hopkins and King 2010; Spirling 2012; Grimmer and Stewart 2013; King, Pan, and Roberts 2013). Other methods extract information or events from text more directly to produce datasets of political events (Schrodt, Davis, and Weddle 1994; Gerner et al. 2002; Schrodt 2012; Beieler et al. 2016). While other forms of text analysis have become increasing multilingual (Lucas et al. 2015), event data has largely remained mono-lingual (though see Osorio 2015; Osorio and Reyes 2017). In this paper, we describe new tools and processes for creating structured event data from unstructured news text. These tools are useful for researchers seeking to make event data in other languages, for extending English-language event data systems to new types of events or actors, and for users of existing datasets to understand the process by which event data dictionaries are developed.

Event data in political science at its most basic consists of a "triple" of information: an event, such as a protest or attack, performed by a source actor against a target. These events and actors are automatically recognized in text, extracted, and resolved to a defined set of codes, such that "demonstrated" and "rallied in the streets" would both be coded as a PROTEST event and "Angela Merkel" and "German Ministry of Defense" would both be represented as DEU GOV. Performing this process on many millions of documents produces a set of structured data that is much easier to analyze than the raw documents.

In producing event data, we build on the dominant paradigm of event coding in English, which consists of automatically comparing grammatically parsed sentence text with hand-defined dictionaries using an event coding tool. The tool follows instructions about how to combine the extracted noun- and verb phrases into a direct event with a source and target, and resolves the extracted text to specified set codes defined in an ontology. An automated event coding system thus consists of two components: a set of dictionaries that map noun and verb phrases to their corresponding actor and event codes in an ontology, and an event coder that applies these dictionaries to the text and makes decisions about how to combine individual actors and actions into coded events.

The event coder we use is UniversalPetrarch[2], an extension of the earlier Petrarch2[3] event coder, specifically built as part of our project to handle multiple languages. To be able to work with multiple languages, UniversalPetrarch takes in articles that have been pre-processed using a dependency parser, which labels the grammatical dependencies between words in a sentences. Much more sophisticated than part-of-speech tagging that labels words as VERB or NOUN a dependency parser determines, for instance, that a particular noun is in fact the subject noun of a verb, while another noun may be that verb's direct object. Handing this grammatical parsing off to a separate program means that UniveralPetrarch can abstract away from a language, allowing it to work on any language that is automatically parsable.[4]

Creating Event Data

The full process of producing events from text consists of three steps. First, the text is put through a natural language processing step, which annotates the sentence with grammatical information about the verb and noun phrases in the text and their relationship.[5] As part of this step we also perform a lemmatizing step that converts different versions of a word into a single "lemma" form. For example, "say", "said", and "says" would all be lemmatized to the simple form "say". This step is convenient in English, but crucial in highly inflected languages such as Arabic.

The second step is to locate potential events in the text, based on the grammatical structure of the text. Using the grammatical information from the previous step, this step, performed with our software UniversalPetrarch, looks for combinations of noun-and verb-phrases that

[2]https://github.com/openeventdata/UniversalPetrarch

[3]https://github.com/openeventdata/petrarch2

[4]See http://universaldependencies.org/

[5]To perform this dependency parsing, we use a custom UDPipe model trained on the Universal Dependencies Arabic data.

are likely to be the actors, targets, and actions in an event.

The third step is also performed by UniversalPetrarch and consists of comparing the extracted actor text and action text to a defined set of phrases and the codes they should receive. For example, if the previous step recognizes "marched and chanted slogans" as an action in the piece of text, this step would resolve it to a consistent, defined event type, such as PROTEST. Similarly, the extracted actor text "Angela Merkel" could be resolved to "DEU GOV". This step thus performs two important functions. First, even after events have been recognized and extracted from text, the sheer variety of terms and language to refer to people, organizations, and events means that raw text phrases are impossible to analyze quantitatively on their own. Resolving them to common codes makes further analysis feasible. Second, because the previous step does not consider the content, only the grammatical structure of the sentence, it can extract events that are not interesting or relevant, such as sports stories or marriage announcements. If their actors or events are not in the dictionary, no actor or event codes will be returned and thus no event will be produced. This helps limit the scope of analysis to only events that are politically relevant.

Dictionary Development

While UniversalPetrarch does not need more than minor changes to accommodate new languages, the dictionaries it uses to map phrases to codes need to be completely re-written for each new language. Creating these dictionaries was the bulk of the work we performed in creating the new event dataset. One of the objectives of this project is to make it easier for future researchers to create their own event data sets. To help with this task, we built several coding interfaces which we describe below.

An alternative approach to creating event data from non-English text is to first machine translate the text into English and then use the existing English-language system to generate events from the translated text. We opted to not use this approach for several reasons. First, while machine translation has improved dramatically in the past two years, it is by no means perfect, and the translations it produces are optimized for humans reading normal text, rather than rule-based machines reading political text.[6] Other considerations were the slowness and expense of machine translation systems, along with the reliance on a third-party web-based system.

We assembled a team of native or fluent Arabic speakers to compile the Arabic-language actor and event dictionaries. We recruited approximately 15 students or recent graduates of the University of Oklahoma's Arabic Flagship Program. We trained each of these students on the event ontology we use, CAMEO (Gerner et al. 2002), as well as the interfaces we describe below. Their work was overseen by a senior native Arabic speaking graduate student or staff member, who could check their work and answer questions about how to classify different actors or event types.

Adding actors

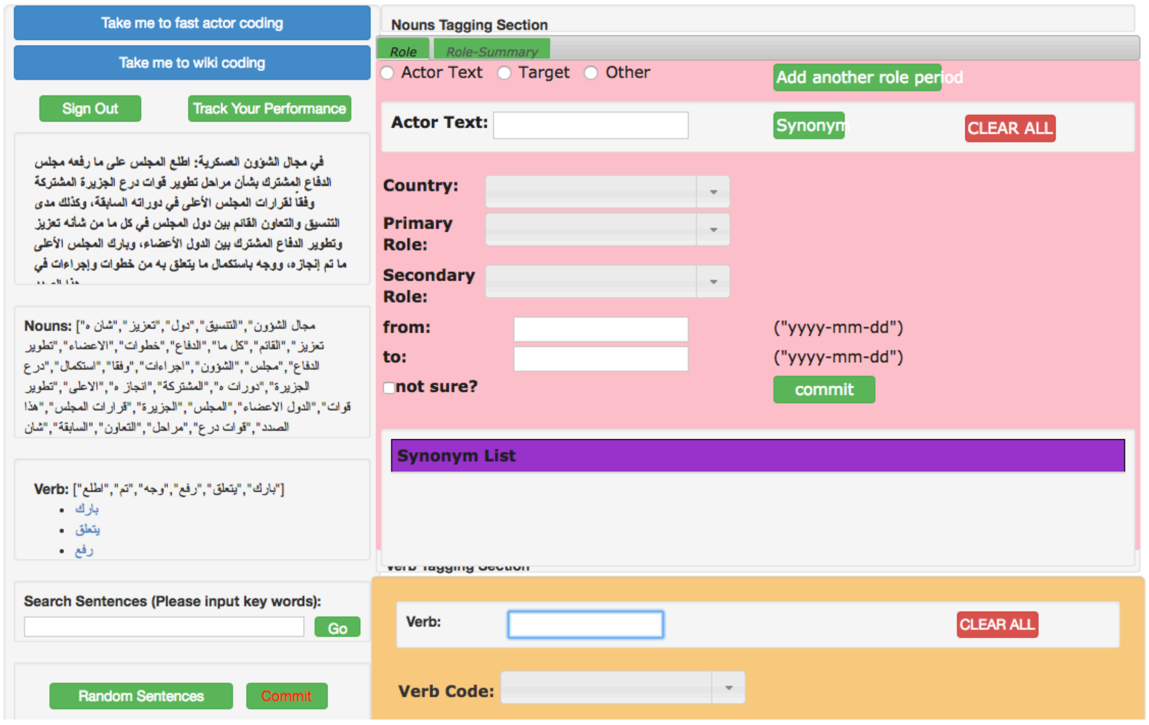
One task in creating event data in a new language is to create the dictionaries that map from actors in the text to a defined set of codes. Recognizing actors in text is not very useful unless the actors are further resolved to a consistent set of codes. For instance, "Chancellor Merkel" and "Angela Merkel" both refer to the same person, who is a member of the German government. In the CAMEO ontology we use, the chancellor of Germany would be represented as "textsc{deu gov}". To perform the mapping from text to code, our system, along with all the current event data systems, relies on predefined "dictionaries" that look for matches in the extracted text and,

[6]A very interesting future study would be to compare the performance of a machine translation event data system with a language-customized system.

if a match is found, returns the code. Actors often change roles over their careers, so actor dictionaries also need to include date ranges for each code. This architecture means that only actors that have an entry in the dictionary will be coded. Because actors in previous event data systems were previously added to the dictionaries by hand, creating actor dictionaries is extremely laborious.

We used several techniques to create the Arabic-language actor dictionaries, ranging from primarily manual to fully automated techniques. We outline each approach, describe the interfaces we use for each, and provide recommendations for other researchers creating non-English language actor dictionaries or researchers developing new ontologies.

The first technique we used to create actor dictionaries was to sample documents from our corpus of text and extract actors from the text using UniversalPetrarch; our team of coders then created a new dictionary entry for each extracted actor. For the purposes of replicability, we used Gigaword for this dictionary development technique. This process replicates the process that Philip Schrodt and colleagues at the University of Kansas used to create the original English language dictionaries. In TABARI, the coding program that Schrodt used, the program would display a single sentence if the existing dictionaries recognized an event but not an actor. Coders would then manually add the actor, if applicable, to the actor dictionary. Our process improves on TABARI's manual approach is several ways. First, rather than TABARI's command line text-based interface, we use a web-based interface that allows coders to select country and role codes from the interface, rather than requiring them to have all country- and role codes memorized. The interface will also suggest alternative versions or spellings of names with the "synonym" button, based on the word2vec (Mikolov et al. 2013) similarity of their usage in the text. This greatly increases the yield of the system, as many versions of the same name can be added at once. The interface also includes an "unsure" button that flags the entry for review by a buddy or the supervisor of the coding process.


Take me to fast actor coding

Take me to wiki coding

Sign Out | Track Your Performance

في مجال الشؤون العسكرية: اطلع المجلس على ما رفعه مجلس
الدفاع المشترك بشأن مراحل تطوير قوات درع الجزيرة المشتركة
وفقاً لقرارات المجلس الأعلى في دوراته السابقة، وكذلك مدى
التنسيق والتعاون القائم بين دول المجلس في كل ما من شأنه تعزيز
وتطوير الدفاع المشترك بين الدول الأعضاء، وبارك المجلس الأعلى
ما تم إنجازه، ووجه باستكمال ما يتعلق به من خطوات وإجراءات في

Nouns: ["مجال الشؤون","التنسيق","دول","تعزيز","شان ه
تعزيز","القائم","كل ما","الدفاع","خطوات","الاعضاء","تطوير
الدفاع","مجلس","الشؤون","اجراءات","وفقا","استكمال","درع
الجزيرة","دورات ه","المشتركة","انجاز ه","الاعلى","تطوير
قوات","الدول الاعضاء","المجلس","الجزيرة","قرارات المجلس","هذا
الصدد","قوات درع","مراحل","التعاون","السابقة","شان

Verb: ["بارك","يتعلق","رفع","وجه","تم","اطلع"]

- بارك
- يتعلق
- رفع

Search Sentences (Please input key words): Go

Random Sentences | Commit

Nouns Tagging Section

Role | Role-Summary

○ Actor Text ○ Target ○ Other | Add another role period

Actor Text: | Synonym | CLEAR ALL

Country:

Primary Role:

Secondary Role:

from: ("yyyy-mm-dd")

to: ("yyyy-mm-dd")

☐ **not sure?** | commit

Synonym List

Verb Tagging Section

Verb: | CLEAR ALL

Verb Code:

Figure 1: Original coding interface. Extracted noun and verb sentences are on the left. Clicking one populates the coding environment on the right, where coders then select the role codes and dates corresponding to the noun, and the event type corresponding to verb phrases.

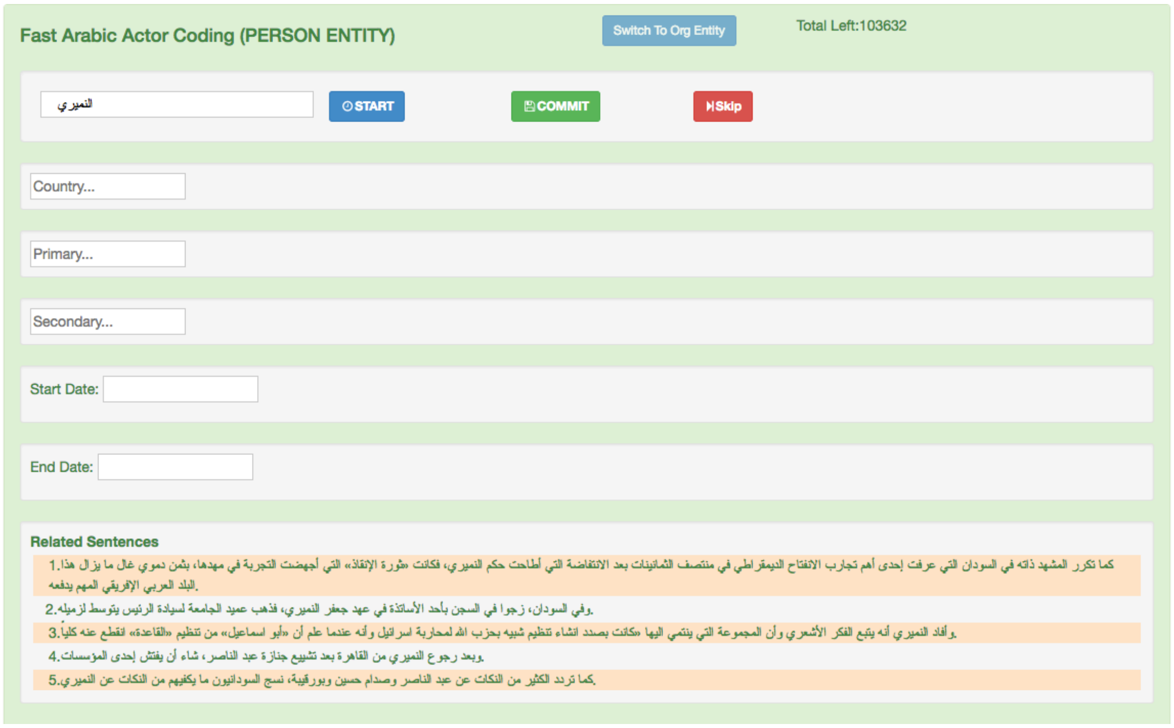
The advantage of this interface is that it actually pulls up actors that occur in the text, ensuring that the dictionaries will have entries for them. The disadvantage of this approach is that we are randomly sampling sentences out of a corpus of millions of sentences, meaning that that coders may be recording actors that are not, in fact, high enough priority to code. We increased the yield from this method by only sampling documents with high proportions of political topics from a topic model we ran on the text, but we are still limited in this method by which documents appear in the sample. We also do not take advantage of information we have from the existing English dictionaries. Using this interface, our coders added 6,387 actor entries to the dictionaries and 1,628 verbs.

The second approach we used in actor coding was to automatically find Arabic transliterations for existing actors in the English dictionary, using Wikipedia. For each actor in the English dictionaries, we attempted to find its Wikipedia page by checking for a page with the person's exact name. When we found an article, we then checked to see if a corresponding Arabic Wikipedia page existed. If so, we would take the actor name in Arabic and the existing role information (dates and labels for positions held) and add them to the Arabic dictionaries. The major advantage of this approach is in its speed and efficiency: no human intervention is required and the actors that have both English and Arabic Wikipedia pages are likely to be the most important. The disadvantages with this approach are that we can only add actors that are already in the English dictionaries, which may be out of date and will not include the full range of relevant actors in Arabic language text. Using this approach, we were able to generate entries for 5,696 actors, more than any other method and with zero marginal cost once we wrote the linking program.

The third approach we used to generate entries for the Arabic actor dictionaries was to automatically find high-frequency actors in the corpus of text (again using Gigaword) and to have coders generate entries for them. This step avoids the rarity problem with the initial approach of randomly selecting sentences. Coders are shown only people and organizations that are mentioned many times in the collection of text but which do not exist in the dictionaries, ensuring that they are always working on the most important previously unadded actors.

Recognizing people and organizations in text is part of a "named entity recognition" (NER) task in natural language processing. Many off-the-shelf programs for recognizing named entities exist, though their performance in Arabic is relatively poor. We initially used a pre-trained multilingual NER model for spaCy[7]. The model is trained on poor data, so its performance is poor, and it does not distinguish between politically relevant and irrelevant people. Out of the most frequent 6,667 people or organizations recognized by the spaCy multilingual model, only 179 were added as actors. This low yield is attributable to the system returning text that is not a named entity, returning names that are politically irrelevant, or names that are too vague (i.e. common first or last names) to be added to the dictionaries. Our coders have annotated several thousand sentences in Arabic to create a customized NER model, which should greatly improve the yield of this approach in the future. The advantage of this approach is that it finds the actors that are most frequently mentioned in the corpus of text we are interested in coding, allowing us to tailor our coding efforts to the most important actors in the corpus, which may not be the most prominent political figures in the region overall. The disadvantage, though, is substantial. Coders expended a large amount of effort skipping through irrelevant actors to find ones to add, which is time consuming and bad for coder morale.

[7]http://spacy.io



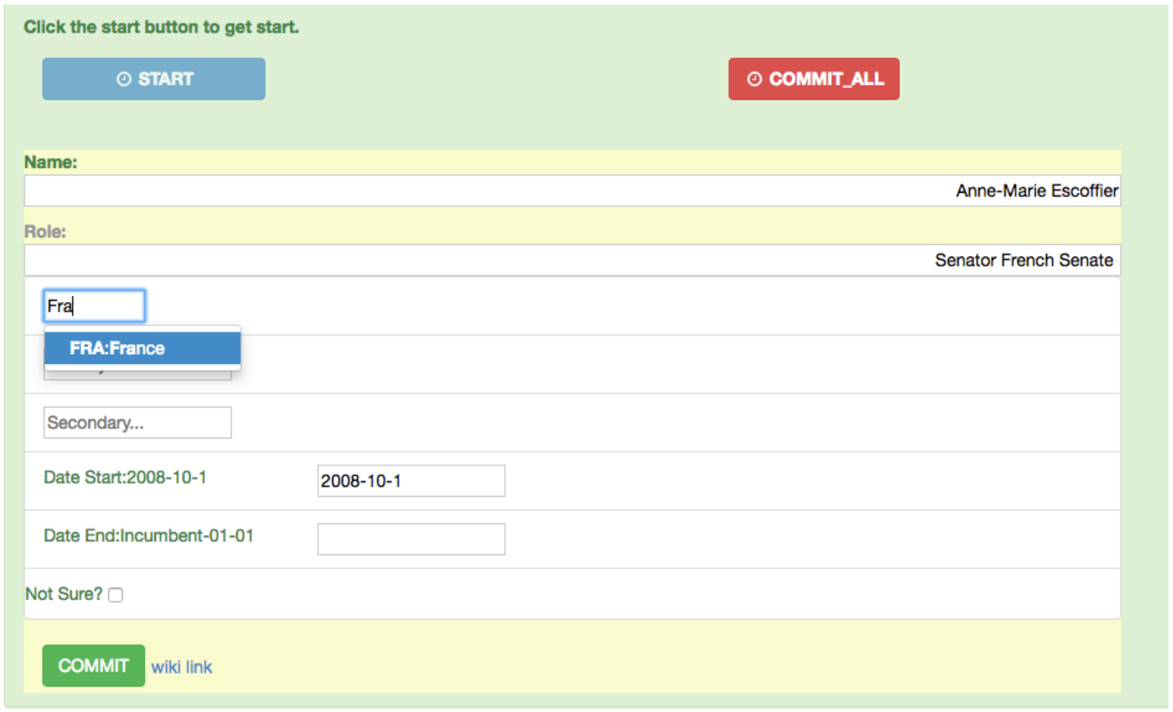
Related Sentences

1. كما تكرر المشهد ذاته في السودان التي عرفت إحدى أهم تجارب الانفتاح الديمقراطي في منتصف الثمانينات بعد الانتفاضة التي أطاحت حكم النميري، فكانت «ثورة الإنقاذ» التي أجهضت التجربة في مهدها، بثمن دموي غال ما يزال هذا البلد العربي الإفريقي المهم يدفعه.
2. وفي السودان، زجوا في السجن بأحد الأساتذة في عهد جعفر النميري، فذهب عميد الجامعة لسيادة الرئيس يتوسط لزميله.
3. وأفاد النميري أنه يتبع الفكر الأشعري وأن المجموعة التي ينتمي اليها «كانت بصدد انشاء تنظيم شبيه بحزب الله لمحاربة اسرائيل وأنه عندما علم أن «أبو اسماعيل» من تنظيم «القاعدة» انقطع عنه كلياً.
4. وبعد رجوع النميري من القاهرة بعد تشييع جنازة عبد الناصر، شاء أن يفتش إحدى المؤسسات.
5. كما تردد الكثير من النكات عن عبد الناصر وصدام حسين وبورقيبة، نسج السودانيون ما يكفيهم من النكات عن النميري.

.

The final approach we used was to scrape Arabic Wikipedia category pages for relevant people (e.g. "Politicians in Iraq"), structure the sidebar information on their careers, and present the extracted information to coders in a web interface. Coders were then asked to provide the correct actor code for each position the person held and to verify the automatically extracted date ranges. Using this approach, we generated entries for 2,327 actors, totaling 4,286 role periods, since many actors hold different positions over time with different corresponding codes. The advantage of this approach is that coders are working only on actors that are politically relevant, because we collect the actors' names from relevant Wikipedia category pages. Coding is also extremely fast: the relevant information is extracted automatically from the Wikipedia page and presented to the coders in a highly structured format, with date information already pre-populated in the page. The disadvantages are that not all relevant actors have Wikipedia pages, nor do these pages always have biographical sidebars. Organizations also do not have biographical sidebars as people do, making this interface useful only for coding people.

The actor dictionaries include a list of generic titles (e.g. "troops", "senator", or "townspeople")



Name:
Anne-Marie Escoffier

Role:
Senator French Senate

Fra
FRA:France

Secondary...

Date Start:2008-10-1 2008-10-1

Date End:Incumbent-01-01

Not Sure? ☐

Figure 2: Interface for coding Wikipedia biography sidebars. The extracted position title is displayed and coders will convert it into its CAMEO representation using structured input boxes.

that are not specific to any country but can still generate a role code (here, MIL, LEG, and CVL, respectively). They also include different versions of country names and their major cities. Because translating the generic titles was a straightforward and structured task, an experienced coder simply translated them directly. To generate the list of country terms, each coder was given a set of instructions and a list of countries and manually created the country entry.

method	total actors coded	total verbs coded
regular interface	6,387	1,628
wiki translation	5,696	NA
NER coding	179 (with 6,667 skipped)	NA
wiki bio coding	2,327	NA

Adding verbs

Besides actors, the second component of the dictionaries is the verb dictionaries that code the event type. An entry in the verb dictionary consists of a verb phrase ("canceled diplomatic visit") and the CAMEO event type it should be coded as ("REDUCE RELATIONS", or the more specific 4-digit numerical code). Events can often be described in much greater diversity than actors can, and the CAMEO framework we use has many more types of events (ca. 250) than types of actors (ca. 20). Both of these make adding verbs to the dictionary a more difficult task than actors. We used two separate interfaces to add verb information to the dictionary, one that extracted verb phrases from the raw text, and one that translated existing entries in English language verb dictionaries to Arabic.

The first interface we developed allows coders to add both actors (described above) and verb phrases to their respective dictionaries. It uses UniversalPetrarch to find verb phrases, which coders then select and assign a CAMEO code to. Verbs on their own rarely convey enough information to be added on their own (canceling a diplomatic visit, military exercise, or an election are all quite different events) meaning that coders usually need to augment the verb with a direct object or prepositional phrases to ensure the correct meaning is captured. Knowing how much detail to include in dictionary entries is one of the important skills that coders need to have: including language that is too specific will lead to an entry that never matches another sentence, making it useless, while a verb entry that does not include important context words will produce erroneous codings, which is even worse. Moreover, the Petrarch2 and UniversalPetrarch coding schemes depend on grammatical knowledge included in the dictionaries. In order to correctly match sentences in production, the event coder needs to know which parts of the entries are direct objects and which are prepositional phrases. Coders thus need to include this information in their entries by marking them with parentheses or braces. Using this interface, coders added 1,628 verb phrases to the dictionary.

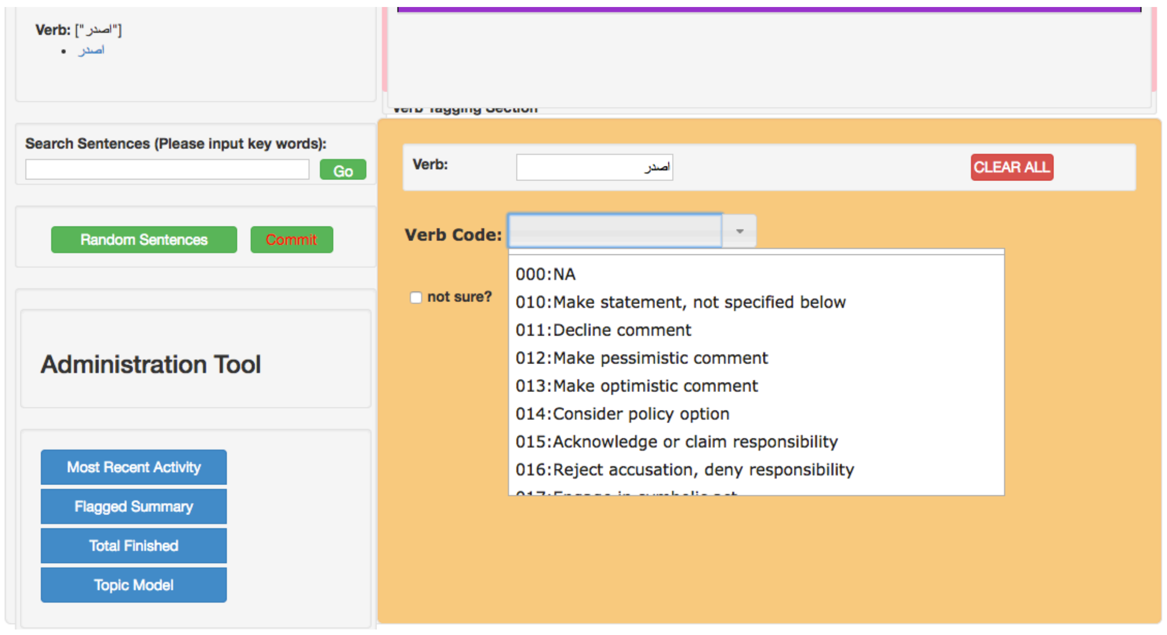

Verb: ["اصدر"]
اصدر
Search Sentences (Please input key words):
Go
Random Sentences
Commit
Administration Tool
Most Recent Activity
Flagged Summary
Total Finished
Topic Model
Verb:
اصدر
CLEAR ALL
Verb Code:
not sure?
000:NA
010:Make statement, not specified below
011:Decline comment
012:Make pessimistic comment
013:Make optimistic comment
014:Consider policy option
015:Acknowledge or claim responsibility
016:Reject accusation, deny responsibility

Figure 3: Original interface for adding verb phrases from text

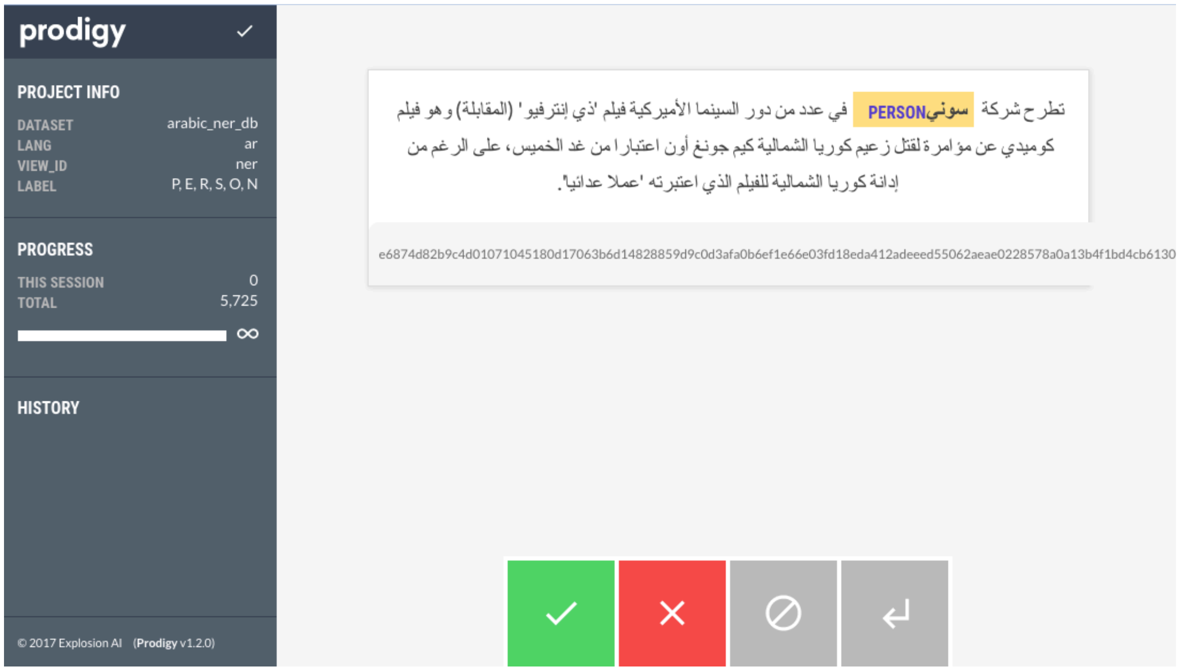
The second approach we took was to use an interface developed by other colleagues on the project that allows the coders to translate the English verb dictionaries into other languages.

They describe the interface and the process of creating Spanish language verb dictionaries in Osorio et al. (2017). The interface provides a structured way for coders to translate each verb and any direct objects or prepositional phrases into the target language (recall that the verb alone rarely provides enough detail to categorize the sentence's event). Specifically, the interface splits these two tasks into two steps. We used a larger group of native Arabic speakers to provide idiomatic translations of direct or indirect objects and prepositional phrases into Arabic, as this requires mastery of both languages. As we discuss in the previous section, the automated event coder also requires the entries in the verb dictionary to include some grammatical markup to help it accurately match sentence text with the dictionaries. The team of coders therefore needed to provide these annotations, which requires a mastery of Arabic grammar. We used a smaller team of fluent Arabic learners to translate just the individual verbs. To speed this second process, the interface provided them a list of machine translations and asked them to mark the correct ones, as well as to add any translations that were not included in the automatically generated list. Each translation for both the verb itself and the larger verb phrase was done by one coder and checked over and corrected if necessary by a second coder.

Using this interface, our team of 6 coders were able to translate the English CAMEO dictionary, with around 9,000 entries, into Arabic in about 5 months of part-time work. This rate works out to about 5 minutes per entry in the verb dictionary, which includes both training, meetings, and checking time. The advantages of this approach are great. By translating the English verb dictionaries that have been developed over decades, we guarantee good coverage over the ca. 250 event categories defined in CAMEO and make our dictionaries roughly comparable with the English dictionaries. It also greatly reduces the time needed to accumulate entries in the verb dictionary, especially because extracting verb phrases by frequency is much more difficult than finding high-frequency actors. The disadvantage of this approach is that it limits the dictionaries to phrases that exist only in English, potentially missing many idiomatic Arabic phrases or phrases that code the kinds of events that are more prevalent in Arabic text.

Named entity recognition

In addition to creating actor and verb dictionaries, the team of coders also created new labeled data to improve Arabic language named entity recognition (NER). Good named entity systems can dramatically improve the process of discovering prominent people and organizations in text that have not been added to the dictionaries, making it much faster to tailor dictionaries to a specific corpus of text. Named entity recognition for place names is also a crucial first step in geolocating events in text (Halterman 2017), which is important for researchers' abilities to study subnational variation in event types. To train the NER system, we used a third-party web interface called Prodigy,[8] which suggests possible labels that coders then accept or reject, making the data annotation process extremely fast. Our coders labeled the named entities in around 6,000 sentences, which is almost as many annotated sentences as there are in the largest public corpus NER-annotated Arabic text, OntoNotes.[9]



PROJECT INFO

DATASET	arabic_ner_db
LANG	ar
VIEW_ID	ner
LABEL	P, E, R, S, O, N

PROGRESS

THIS SESSION	0
TOTAL	5,725

تطرح شركة سوني PERSON في عدد من دور السينما الأميركية فيلم 'ذي إنترفيو' (المقابلة) وهو فيلم كوميدي عن مؤامرة لقتل زعيم كوريا الشمالية كيم جونغ أون اعتبارا من غد الخميس، على الرغم من إدانة كوريا الشمالية للفيلم الذي اعتبرته 'عملا عدائيا'.

e6874d82b9c4d01071045180d17063b6d14828859d9c0d3afa0b6ef1e66e03fd18eda412adeeed55062aeae0228578a0a13b4f1bd4cb6130

Figure 4: Using Prodigy to train a named entity recognition system

[8]https://prodi.gy/
[9]https://catalog.ldc.upenn.edu/ldc2013t19

Conclusions and Recommendations

Using these techniques for developing dictionaries, we were able to complete Arabic actor and verb dictionaries with coverage equivalent to the English language dictionaries in less than two years of work, compared to the two decades that the English language dictionaries took to produce. We have used UniversalPetrarch to generate events from our corpus of millions of Arabic language sources and expect to make the dataset and comparisons between it and an English language corpus available after final debugging and quality checking.

We would like to draw conclusions from our process of making dictionaries for event data coding to help researchers who may produce their own dictionaries in the future. We believe that the tools we've developed over the past two years and the techniques for dictionary development that we've learned can reduce development time even further. The advice that we have applies both to researchers interested in translating the existing CAMEO dictionaries from English into a new language, but also researchers developing dictionaries for wholly new event or actor types in English or another language.

In developing actor dictionaries, we recommend starting by linking the existing English language actor dictionaries with Wikipedia and using Wikipedia to provide translations into other languages. Because we have already performed the linking step and made our dictionaries and code available, researchers making dictionaries for a new language will need to do very minimal work to get several thousand important people and organizations in their target language. The second step we recommend is to scrape Wikipedia category pages for actors of interest, for both translating CAMEO into a new language and for implementing wholly new ontologies. This process gives good coverage of actors and in the case of people with biographical sidebars, permits extremely rapid coding. Finally, we recommend extracting actors from the target text using UniversalPetrarch or using named entity recognition and coding the most frequent actors. This step should be performed after the previous two, because our interfaces will skip actors it already has in the dictionaries. Finally, we do not recommend the original approach of coding actors from randomly selected sentences. Doing so has the advantage of increasing the breadth and diversity of actors coded, but most actors that appear using this approach will be unimportant. Efforts would better be spent on the other three approaches.

In translating verb dictionaries, we highly recommend using the CAMEO verb translation interface described in Osorio et al. (2017). It very quickly provides coverage that is similar to the existing English dictionaries (for better or worse) in a structured way that works well with student coders. It does not, however, work for researchers who are developing wholly new event categories to study events specific to their research. In this case, or for researchers who are interested in expanding the coverage of particular CAMEO categories, we recommend the first interface we developed, with a topic model selection process. To make sure our coders were generating dictionary entries from politically relevant news stories, we ran a topic model over our entire corpus of text. We then decided which topics were relevant to our work, and only sampled sentences with high proportions from those topics, using functionality built into the interface. For targeted verb development, we recommend running a series of topic models until topics appear that are related to the event type of interest. Sentences to code should then be drawn from those topics to maximize the chances of coders encountering sentences that contain the event of interest.

One of the great promises of this new era of automated text analysis in political science is the ability of researcher to produce new datasets that would have been impossible to create with previous technology. Off-the-shelf datasets, including the Arabic event dataset we have produced will play an important role in furthering research, but potentially more insights will come from researchers developing their own specific datasets for answering their own questions. We hope that the tools we have built and the knowledge we have gained through this process will help future researchers in making their own event datasets.

References

Beieler, John, Patrick T Brandt, Andrew Halterman, Erin Simpson, and Philip A Schrodt. 2016. "Generating Political Event Data in Near Real Time: Opportunities and Challenges." In *Data Analytics in Social Science, Government, and Industry*, edited by R. Michael Alvarez. Cambridge University Press.

Gerner, Deborah J., Philip A Schrodt, Omür Yilmaz, and Rajaa Abu-Jabr. 2002. "Conflict and Mediation Event Observations (CAMEO): A New Event Data Framework for the Analysis of Foreign Policy Interactions." *International Studies Association, New Orleans*.

Grimmer, Justin, and Brandon M Stewart. 2013. "Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts." *Political Analysis* 21 (3). Oxford University Press: 267–97.

Halterman, Andrew. 2017. "Mordecai: Full Text Geoparsing and Event Geocoding." *The Journal of Open Source Software* 2 (9). The Open Journal. doi:10.21105/joss.00091.

Hopkins, Daniel J, and Gary King. 2010. "A Method of Automated Nonparametric Content Analysis for Social Science." *American Journal of Political Science* 54 (1). Wiley Online Library: 229–47.

King, Gary, Jennifer Pan, and Margaret E Roberts. 2013. "How Censorship in China Allows Government Criticism but Silences Collective Expression." *American Political Science Review* 107 (2). Cambridge University Press: 326–43.

Lucas, Christopher, Richard A Nielsen, Margaret E Roberts, Brandon M Stewart, Alex Storer, and Dustin Tingley. 2015. "Computer-Assisted Text Analysis for Comparative Politics." *Political Analysis*, mpu019.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. "Distributed

Representations of Words and Phrases and Their Compositionality." In *Advances in Neural Information Processing Systems*, 3111–9.

Osorio, Javier. 2015. "The Contagion of Drug Violence: Spatiotemporal Dynamics of the Mexican War on Drugs." *Journal of Conflict Resolution* 59 (8). SAGE Publications Sage CA: Los Angeles, CA: 1403–32.

Osorio, Javier, and Alejandro Reyes. 2017. "Supervised Event Coding from Text Written in Spanish: Introducing Eventus ID." *Social Science Computer Review* 35 (3). SAGE Publications Sage CA: Los Angeles, CA: 406–16.

Osorio, Javier, Viveca Pavon, Jennifer S. Holmes, Sayeed Salam, and Patrick T. Brandt. 2017. "Translating CAMEO: Documenting the Translation into Spanish." *Prepared for the 75th Annual Meeting of the Midwest Political Science Association, April 6–9, 2017.*

Schrodt, Philip A. 2012. "Precedents, Progress, and Prospects in Political Event Data." *International Interactions* 38 (4): 546–69.

Schrodt, Philip A, Shannon G Davis, and Judith L Weddle. 1994. "Political Science: KEDS—a Program for the Machine Coding of Event Data." *Social Science Computer Review* 12 (4). Sage Publications Sage CA: Thousand Oaks, CA: 561–87.

Spirling, Arthur. 2012. "US Treaty Making with American Indians: Institutional Change and Relative Power, 1784–1911." *American Journal of Political Science* 56 (1). Wiley Online Library: 84–97.