2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)

Automatic Event Coding Framework for Spanish Political News Articles

Sayeed Salam
Department of Computer Science
The University of Texas at Dallas
Richardson, Texas, United States
sxs149331@utdallas.edu

Lamisah Khan
Plano West Senior High School
Plano, Texas, United States
parveen.pallabi@gmail.com

Amir El-Ghamry
Department of Computer Science
Mansoura University
Mansoura, Egypt
amir_nabil@mans.edu.eg

Patrick Brandt, Jennifer Holmes, Vito D'Orazio
School of Economic, Political, & Policy Sciences
The University of Texas at Dallas
Richardson, Texas, United States
{pbrandt, jholmes, dorazio}@utdallas.edu

Javier Osorio
School of Government and Public Policy
University of Arizona
Tucson, Arizona, United States
josorio1@email.arizona.edu

***Abstract*—Today, Spanish speaking countries face widespread political crisis. These political conflicts are published in a large volume of Spanish news articles from Spanish agencies. Our goal is to create a fully functioning system that parses realtime Spanish texts and generates scalable event code. Rather than translating Spanish text into English text and using English event coders, we aim to create a tool that uses raw Spanish text and Spanish event coders for better flexibility, coverage, and cost.**

To accommodate the processing of a large number of Spanish articles, we adapt a distributed framework based on Apache Spark. We highlight how to extend the existing ontology to provide support for the automated coding process for Spanish texts. We also present experimental data to provide insight into the data collection process with filtering unrelated articles, scaling the framework, and gathering basic statistics on the dataset.

***Index Terms*—Multilingual, Apache Spark, Universal Dependency, Automated Event Coder, NLP, BERT**

I. Introduction

Spanish is the second most spoken language in the world with over 460 million native speakers. Out of a total of 195 countries, there are twenty Spanish-speaking countries, such as Spain, Venezuela, Colombia, Mexico, etc. Spanish-speaking countries have become a hotbed for political conflict like the crisis of leadership in Venezuela or Colombia's war on drugs. With the prominence of Spanish-speaking countries in global interactions and political instability ravaging these countries, it becomes pertinent to create a processing tool that can parse through Spanish news for signs of political crisis. This tool will parse unstructured Spanish text into structured event code.

Automated coders (i.e. PETRARCH) are specifically designed to complete structured interpretation of political events in English and are guided by ontologies like CAMEO, a Conflict and Mediation Event Observation ontology [1]–[3]. This ontology is laid out in such a way that the automated event coder can generate Source-Action-Target (SAT) formatted events where "Source" is the initiator, "Action" is what has been initiated, and "Target" is the entity being acted upon. The format is also known as the *who-did-what-to-whom* pattern.

However, PETRARCH is limited in that it only works for English news articles. It uses language specific parsers (i.e. Stanford CoreNLP [4]) to generate parse trees and works with relationships between words to find useful patterns representing actions and named entities. The process is hard to extend in other languages without rewriting the core logic from ground up. This problem originates due to the non-uniform nature of metadata (i.e., parts of speech tags) across different languages.

Although, substantial work has been done to convert unstructured English text to structured event code, there has been little to no work done to convert Spanish unstructured text to structured event code.

We develop a tool, the first of its kind, that facilitates structured political event code from unstructured Spanish news articles. Event coding in Spanish can be done in a number of ways. The straightforward way is to translate Spanish text into English and perform English event coding on the translated texts. The drawbacks of this would be cost and the poor quality of translation, hindering the tool's performance. Another way is to develop a tool that will work with the raw Spanish text and perform Spanish event coding. We adopt this later strategy. To event code in Spanish, we cannot use readily-available CAMEO ontology, because CAMEO ontology defines verb-action patterns and actors in English. So, instead, we create an extended ontology that contains the verb-action patterns and actors in Spanish. Using this extended ontology, we parse Spanish texts with a language independent parser called UDPipe and match words with event code.

To facilitate event coding for Spanish texts, we make the following contributions: first, we extend cameo ontology for verb-action patterns and actors for Spanish texts; second, we utilize language independent parsers; third, we develop a full fledged framework that captures a stream of real time Spanish

978-1-7281-6873-9/20/$31.00 ©2020 IEEE

DOI 10.1109/BigDataSecurity-HPSC-IDS49724.2020.00052

texts from various Spanish news websites using Apache Spark [5] and Kafka [6], parses them, and generates automated event code; finally, we develop a fully functional prototype and empirically analyze its effectiveness.

The paper is organized as follows. Section II highlights some key concepts and tools that help the reader understand the rest of the paper. Section III highlights the organization of different modules in the pipeline. Section IV shows results obtained on the performance of different modules. Section V represents the current research and development work going into to our system. Finally, Section VI presents the concluding remarks along with the future direction of our work.

II. Background

To help the reader better understand the tools and methods used in this paper, we provide key details of fundamental concepts.

Conflict and Mediation Event Observations (CAMEO) [1] is an ontology developed to capture political events and focuses primarily on the following four categories, also known as ”Quad Classes”.

- Verbal Cooperation - when two entities are agreeing or co-operating verbally on a matter
- Material Cooperation - when an entity (source) is actively helping another entity (target)
- Verbal Conflict - when two entities are in disagreement on a matter
- Material Conflict - when an entity (source) is in conflict physically with another entity (target), i.e., protest, war, etc

It works using a knowledge-base of pattern and actor dictionaries. The dictionaries contain over 6000 patterns representing 200 types of events (encoded in three digit format like, 010, 370, etc). Pattern dictionaries are helpful for identifying political interactions in a given sentence. Actor dictionaries are used to search for political actors around the matched pattern.

Political Events are structured pieces of information consisting of an action, source (acting entity), target (entity being acted upon) and other related information. For example, consider the following extract from a news article -

```
As coronavirus cases crop up across the
United States, some governors and other
leaders are scrambling to slow its spread,
banning large public gatherings, enforcing
quarantines and calling National Guard
troops.
```

As a structured event, it looks like the following

```
Source - ---GOV
Target - ---MIL
Action - 041 (Discuss by telephone)
```

”governors” and ”National Guard troops” are coded in standard ISO-3 coding format for event generation. The structure used here is mostly known as the *who-did-what-to-whom* format of event coding. The coding mechanism is depicted in the Figure 1

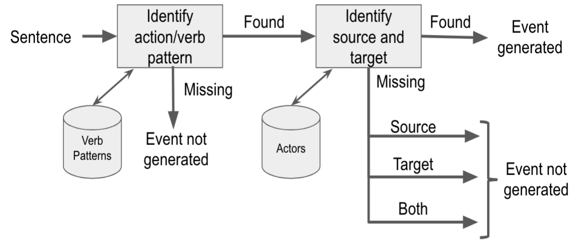


Fig. 1: Basic Mechanism of Automated Coding using PETRARCH

Given a sentence, the encoder searches for a matching pattern in the CAMEO [1] verb patterns dictionary. A pattern consists of a verb and surrounding keywords. The pattern signifies a particular course of action and is represented by event code. For example, SET in the pattern ”SET OUT VIEWS” indicates a MAKE PUBLIC STATEMENT type of event (event code 010). Upon finding a match in pattern, actor dictionaries search for matching entities representing the source and target. After finding the necessary information, an event is coded by PETRARCH. If there is missing information, PETRARCH will ignore the event. This sequence of events are called Source-Action-Target or SAT format.

Universal Dependency (UD) [7] is an technique that supports cross-linguistically consistent tree-bank annotation. Its overall goal is to provide a universal collection of categories and guidelines to facilitate consistent annotation of similar constructions across languages, while allowing language-specific extensions when necessary. We will be using ”ufal-udpipe” [8], a python package for generating universal dependency parse trees.

III. Multilingual Framework

We design an infrastructure to download articles from the web, process them to generate metadata, and run event coding and geolocation algorithms followed by a distribution of the data using a web based API. The following subsections will describe each stage in detail from the framework.

Figure 2 depicts the steps of the framework and how the pipeline works. The framework can be divided into sequential steps [9] as follows -

- Step 1: Data Collection
- Step 2: Preprocessing
- Step 3: Event Coding
- Step 4: Data Access

At Step 1, we collect Spanish news articles using Web Crawler. Then we filter out the Spanish articles and keep only the politically relevant articles using ML based filtering classifier (Filter). Filtered documents are passed through the next step (Step 2) where within the text Processing module universal dependency parse trees are generated at the sentence level (Text Processing). This step runs on Apache Spark to

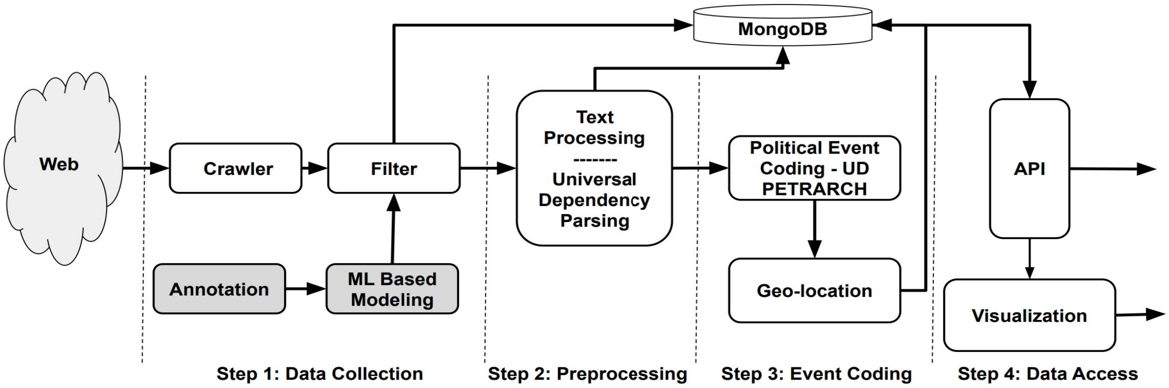


Fig. 2: Multilingual Event Coding Framework Diagram

ensure scalability. The processed output is provided to the Political Event Coding module where UD-PETRARCH (Political Event Coding) and geolocation software work together to generate geolocated events (events along with the place where it happened) and completes Step 3. In the next step, the generated events are then distributed using the API to the Researchers and Visualization tools (API and Visualization). The steps highlighted in Figure 2 represents a real-time streaming data processing unit. There are other components that supports the online framework (module with highlighted background in Figure III). Those are as follows -

- Model building and validation for the news article filter to support ”Step 1: Data Collection” part
- Ontology translation for supporting the ”Step 3: Event Coding”
- Validation of multilingual event coding w.r.t events generated in English.

A. Step 1: Data Collection

In this step, we run a crawler to obtain Spanish news articles using a list of news agencies from Spain, South America, Central America, and other Spanish-speaking countries. These articles that are downloaded daily become the input for the framework. A detailed list of Spanish news websites can be found here [10].

Article Annotation

This step is required for Spanish news articles as we download these articles without topic restrictions and the websites often do not present a way to collect data section wise. Sometimes, a news article may also have URLs to other news articles that the crawler can follow and download. After the data collection, we will get a lot of articles that are not relevant and of those that are relevant, not all of them are political. We annotate a subset of 450+ randomly chosen articles and annotate whether the articles are relevant, and if so, whether they are focused on politics. Once an article is found to be politically relevant, we classify it into quad-class categories of the CAMEO ontology and thus create a simplified Gold Standard Records (GSR). This set of articles will be used for validating the accuracy/coverage of the event coder.

Article Filtration - Relevant/Irrelevant followed by political/non political

We have designed a machine learning algorithm to filter out articles that are not relevant to politics (either irrelevant articles or articles from other sections like business, sports, etc). We have used TF-IDF on Bi-grams for feature selection technique and RandomForest, NaiveBayes, and SVM as classifiers [11]. We also applied deep neural network based classifiers like, BERT [12] with its own feature extraction technique.

B. Step 2: Preprocessing

After filtering out the irrelevant articles, we process the remaining documents to generate metadata from raw text. The metadata includes Parts-of-Speech (POS) tags, named-entity tags along with the dependency relationships. We use ufal-udpipe python package to generate the tags and relationship between words at a sentence level. To explain Step 2 and 3, we use Sentence 1 for reference.

Sentence 1: *The UN Security Council on Tuesday unanimously approved a United States’ resolution on the recent deal between the U.S. and the Afghan Taliban, a rare endorsement of an agreement with a militant group.* [1]

The Spanish translation of Senetence 1 is below -

El martes, el Consejo de Seguridad de la ONU aprobó por unanimidad una resolución de Estados Unidos sobre el reciente acuerdo entre Estados Unidos y los talibanes afganos, un respaldo poco frecuente de un acuerdo con un grupo militante.

For each sentence in a document, we generate a dependency parse tree (Figure 5). The parsing job requires a lot of time to complete when ran in standalone mode. We have adapted a Spark-based distributed system for this task as it speeds up the processing almost linearly with an increase in the number

[1] https://www.thehindu.com/news/international/un-security-council-endorses-us-taliban-deal/article31035580.ece

Noun Phrase	Verb Phrase
Council, Tuesday, U.S.	approved
Afgan, Taliban, resolution	

TABLE I: Noun and Verb Phrases for Sentence 1

of processing cores. Each Spark worker node generates a dependency parse for sentences from a batch of news articles and stores them in a MongoDB instance.

C. Step 3: Event Coding

After generating metadata, we process it with the UD-PETRARCH event coder [13] to generate time-stamped political events, making them CAMEO compatible events. For geolocating events, we use Cliff-Clavin [14] from MediaMeter. It gives us locations associated with the events found in the sentences. Here is a brief description on how the event coding process works for a particular sentence. Given the sentence with dependency relations, the coder first identifies the noun and verb phrases of the sentence. Examples of the phrases are listed in Table I generated from Sentence 1.

The coder then identifies the root verb, which is ”approve” in this case. Then using the dependency relationship between noun phrases and the root verb, the coder creates triplets in the form of (source, action, target). An example triplet that eventually qualifies for an event is as follows:

```
u'matched_txt': u'- * + OF [080]
#line:9440',
u'source_text': u'THE UN SECURITY COUNCIL',
u'target_text': u'A UNITED STATES'
RESOLUTION',
u'verb_text': u'APPROVE',
u'verbcode': u'080'
```

Here the ”matched_text” is the verb pattern in the sentence that has been found in the CAMEO verb dictionary. The source and target are matched against the actor dictionaries and once all the information are found, the triplet becomes an event.

All the processed text articles and generated events are stored in the MongoDB. All these framework components use Apache Kafka to synchronize the inputs and outputs.

Ontology Translation From English

To capture events in articles published in Spanish or in another foreign language, we have to translate the existing CAMEO ontology to the corresponding language. There are two parts that needs to be translated at this step: the verb-patterns and the list of political entities (political leaders, organizations, etc.).

Translating verb patterns: Verb patterns are used to capture the interaction between two entities. To translate verb-patterns, we adopt a semi automated way and develop an online application to facilitate the collaboration between human annotators. We first present a basic translation of verbs using the Wordnet [15] synsets in Spanish. Human annotators are asked to assess the quality of translation w.r.t the CAMEO code category and after feedback, we do a majority analysis to include the translation in the new dictionary [16]. A snippet of translated verb dictionaries are presented in Figure 3

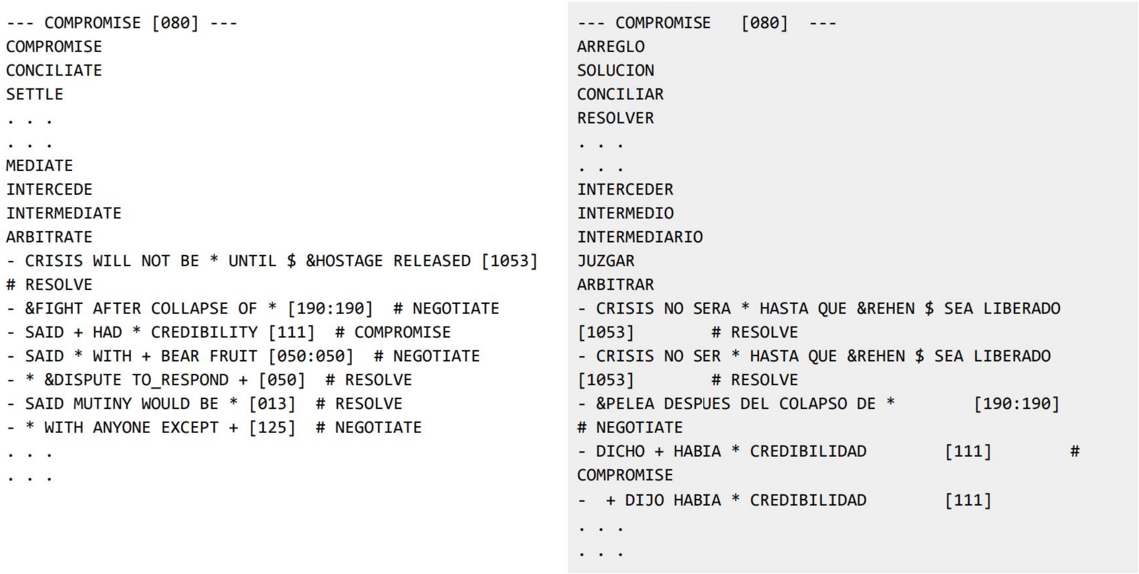
Translating Actors: With this approach we translate CAMEO dictionaries containing political entities and organizations. The algorithm uses BableNet translations database to translate dictionaries written in one language to another specified language. BabelNet is a multilingual encyclopedic dictionary which was created by seamless integration of the largest multilingual Web encyclopedia - i.e., Wikipedia - the most popular computational lexicon of English - i.e., WordNet, and other lexical resources such as OmegaWiki and the Open Multilingual WordNet. We observed that the Bablenet translation database is more accurate for translating agents, actors, countries, and organization names than other translation sources such as Google and JRC databases [17].

Since BableNet is based on multiple lexical resources, like Wiktionary, OmegaWiki, Wikidata, Wikipedia infoboxes, free-license wordnets, Wikiquote, FrameNet, VerbNet, and others, it is able to overcome problems that may arise from other translators such as ambiguous and non-existing translations. Also, BableNet fills in lexical gaps in resource-poor languages with the aid of Statistical Machine Translation and it connects concepts and named entities in a very large network of semantic relations.

The translation process (depicted in Figure 4) takes a dictionary file in English as an input and produces a translated version in Spanish. It goes through each of the entities and their synonyms and translate them using BabelNet database [18] to Spanish. We get several translations and each of them are associated with scores. Translations with the highest scores are considered first. If no translation on BabelNet is present, Google Translate is used to do the translation task. For example, for translating the entity ”AFGHANISTAN”, the translation
”INVASION_DE_AFGANISTÁN_DE_2001”
comes first , while
”ESTADO_ISLAMICO_DE_TRANSICIÓN
_DE_AFGANISTAN” comes second, and ”INVASIÓN_DE_AFGANISTAN_DE _2001” comes in third place based on the associated score.

In the case of translating a synonym, the returned translated synonym set is clustered with other translated synonym sets belonging to the same main Entity. The algorithm leverages Levenshtein distances technique [19] to calculate the similarity between each synonym pair in the synonym set. Then the algorithm creates a two-dimensional array that stores the Levenshtein distance between each pair of synonyms in the set. After that, the algorithm builds a tree structure diagram of possible clustering based on hierarchical clustering techniques [20], [21]. After that, the elbow method [22] is applied to determine the optimal number of clusters based on the hierarchical tree. For example, if the translated synonym set is of size 8 as follows:

```
S1: 'ESTADO_ISLAMICO_DE_TRANSICION_DE
_AFGANISTÁN',
```



```
--- COMPROMISE [080] ---
COMPROMISE
CONCILIATE
SETTLE
. . .
. . .
MEDIATE
INTERCEDE
INTERMEDIATE
ARBITRATE
- CRISIS WILL NOT BE * UNTIL $ &HOSTAGE RELEASED [1053]
# RESOLVE
- &FIGHT AFTER COLLAPSE OF * [190:190]  # NEGOTIATE
- SAID + HAD * CREDIBILITY [111]  # COMPROMISE
- SAID * WITH + BEAR FRUIT [050:050]  # NEGOTIATE
- * &DISPUTE TO_RESPOND + [050]  # RESOLVE
- SAID MUTINY WOULD BE * [013]  # RESOLVE
- * WITH ANYONE EXCEPT + [125]  # NEGOTIATE
. . .
. . .
```

```
--- COMPROMISE   [080]  ---
ARREGLO
SOLUCION
CONCILIAR
RESOLVER
. . .
. . .
INTERCEDER
INTERMEDIO
INTERMEDIARIO
JUZGAR
ARBITRAR
- CRISIS NO SERA * HASTA QUE &REHEN $ SEA LIBERADO
[1053]        # RESOLVE
- CRISIS NO SER * HASTA QUE &REHEN $ SEA LIBERADO
[1053]        # RESOLVE
- &PELEA DESPUES DEL COLAPSO DE *        [190:190]
# NEGOTIATE
- DICHO + HABIA * CREDIBILIDAD        [111]        #
COMPROMISE
-  + DIJO HABIA * CREDIBILIDAD        [111]
. . .
. . .
```

Fig. 3: Snippets from English and Spanish (highlighted) verb dictionaries. The entry starts with a main verb, followed by related verbs and patterns (lines starting with ”-”)

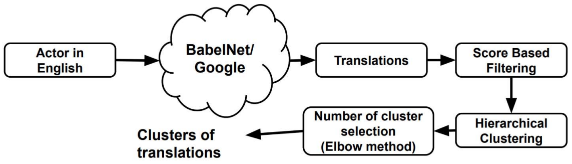


Fig. 4: Steps in translating Actors in English to Spanish

	S1	S2	S3	S4	S5	S6	S7	S8
S1	0	1	2	25	28	31	30	29
S2	1	0	3	25	28	31	30	29
S3	2	3	0	24	29	30	31	30
S4	25	25	24	0	12	14	15	14
S5	28	28	29	12	0	16	19	18
S6	31	31	30	14	16	0	6	5
S7	30	30	31	15	19	6	0	1
S8	29	29	30	14	18	5	1	0

TABLE II: Distances between translated texts identified as S1 to S8

```
S2: ’ESTADO_ISLÁMICO_DE_TRANSICION_DE
_AFGANISTÁN’,
S3: ’ESTADO_ISLAMICO_DE_TRANSICIÓN_DE
_AFGANISTAN’,
S4: ’EJERCITO_DE_AFGANISTAN’,
S5: ’EJERCITO_NACIONAL_AFGANO’,
S6: ’PROVINCIA_DE_PARWAN’,
S7: ’PROVINCIA_DE_BĀMIYĀN’,
S8: ’PROVINCIA_DE_BAMIYĀN’
```

And they are numbered as S1 to S8 Then the two dimensions array of the distances are listed in Table II:

Then, a tree structure diagram of possible clustering is built based on hierarchical clustering techniques, After that the elbow method is applied and result in three clusters based on hierarchical clustering as follows:

```
Cluster 1:
’ESTADO_ISLAMICO_DE_TRANSICION_
DE_AFGANISTÁN’,
’ESTADO_ISLÁMICO_DE_TRANSICION_
DE_AFGANISTÁN’,
’ESTADO_ISLAMICO_DE_TRANSICIÓN_
DE_AFGANISTAN’

# Cluster 2
’EJERCITO_DE_AFGANISTAN’,
’EJERCITO_NACIONAL_AFGANO’

# Cluster 3
PROVINCIA_DE_PARWAN
’PROVINCIA_DE_BĀMIYĀN’,
PROVINCIA_DE_BAMIYĀN
```

Multilingual Event Coding in Spanish

We are using UD-PETRARCH event coder that works on universal dependency rather than only the Parts-Of-Speech tags used by original version of PETRARCH. It is helpful for multilingual event coding because of the uniformity of universal dependency parses across language in comparison to the inconsistency of Parts-of-Speech tags across languages. Using a universal dependency based event coder, we can easily incorporate coding in other languages with no effort needed on updating the event coder itself. We will still need the dictionaries to be translated first. Using a universal dependency parser, however, gives flexibility for the non-CS background researchers to solely focus on the analysis and ontology extension parts.

Figure 5 shows how the dependency relations are structured between words in the translated version of Sentence 1.

Once again the root verb here is ”aprobó”, meaning ”approve” in English. This time UD-PETRARCH genrates the following triplet

```
'matched_txt': '- * + OF [080]
#line:9440',
'source_text': 'EL CONSEJO DE
SEGURIDAD DE LA ONU',
'target_text': 'A ESTADOS
UNIDOS RESOLUCIÓN',
'verb_text': 'aprobó',
'verbcode': '080'
```

and we find the same event where the source is USA, the target is IGOUNO, and the event type code is 080.

Cross-lingual validation for generated events.

To identify whether similar events reported in different languages can be captured by the event coder, we run the following validation procedure. First, we select a set of documents in Spanish that has been annotated to be politically relevant. Then we translate them to English using Google Translate API. Afterwards we have a parallel corpus of English and Spanish documents. Then we run respective language parsers to generate universal Dependency relationships and feed them to the UD-PETRRACH event coder. We observe the generated triplets and do a semantic comparison between reported source and target text with the help of BabelNet [18]. We also follow whether they are reporting the same event type code. We will highlight the findings in Section IV.

D. Step 4: Data Access

We provide API to serve generated event data. This API maintains API-key based access restrictions and serves the data in JSON format. Interested users can query using a JSON based query language (similar to MongoDB). Users can select the portion of the data they are interested in by subsetting the data with query parameters. They can also focus on particular fields on each record, aggregate/group the query results, and query for the metadata about the datasets. Additionally, there are user defined libraries in R programming language [23] built on top of the web-based API to make accessing the data easier. Details of the access policy can be found here [24].

IV. Experiments

In this section we will discuss about the experiments conducted for different modules of the framework.

Scalability: Universal Dependency Parse generation

As we point out earlier, the universal dependency parser is the most compute-intensive task in the pipeline. We adopt a distributed system based on Apache SPARK and Kafka to parse multiple documents in parallel. Each worker node in the Spark cluster gets a subset of the documents to process. The synchronization is maintained by Kafka, guaranteeing that there is no duplicate in processing task. Figure-6 show the relationship between the execution time and number of processing cores and follows a linear monotonic decreasing function. For this experiment we selected a subset of 1,400 Spanish news articles and generated dependency parses for each of the sentences in those articles.

Number of news articles	132
Number of events generated(English)	107
Number of events generated(Spanish)	98
Number of events matched exactly	78
Number of events matched in Event code	93

TABLE III: Comparison between English and Spanish Event coding on parallel corpus.

Document Translation vs Ontology Translation

With our current approach, we translate the CAMEO ontology to support the event coding framework for Spanish news articles. Another route was to translate the articles into English first and then apply an English event coder to generate the events. However, the cost to use paid APIs, like Google Translate for that which would have cost us $125,000 for the current corpus of 2.5 Million Spanish news articles. Also, the cost would have increased with time as the corpus gets larger. This calculation is based on the estimate by Google who states $0.05 is required to translate a document with around 500 characters [25].

Event coding coverage across languages

In this part of the experiment, we run the event coders in parallel for a set of Spanish articles and the English translated version of those articles. We observe the similarities between the events from the original and the translated version. Statistics are shown in the Table-III. As we observed, a large number of events matched exactly with each other. Of the remaining ones, there were partial matches present (ex. USAGOV captured in English as source, GOV in Spanish for the corresponding event).

Article Filtration using ML Classifier

Using the annotated 4̃50 documents, we create a multiclass classifier that tags each document as irrelevant, political and non-political. We observe an overall accuracy of 82.5% with Random Forest classifiers (Table IV) and average accuracy of 67% among the three classes. In our observation, we found that the classifier struggles to differentiate between political and non-political news articles. Most of the irrelevant articles (i.e. ads, homepages, etc.) were identified with reasonable accuracy with no false negatives (articles identified as irrelevant but are actually political/non-political). We also applied Deep Neural Network (DNN) model BERT [12] but found it less accurate ($\sim 75\%$) than the traditional machine learning model, due to less annotated data to train the DNN model.

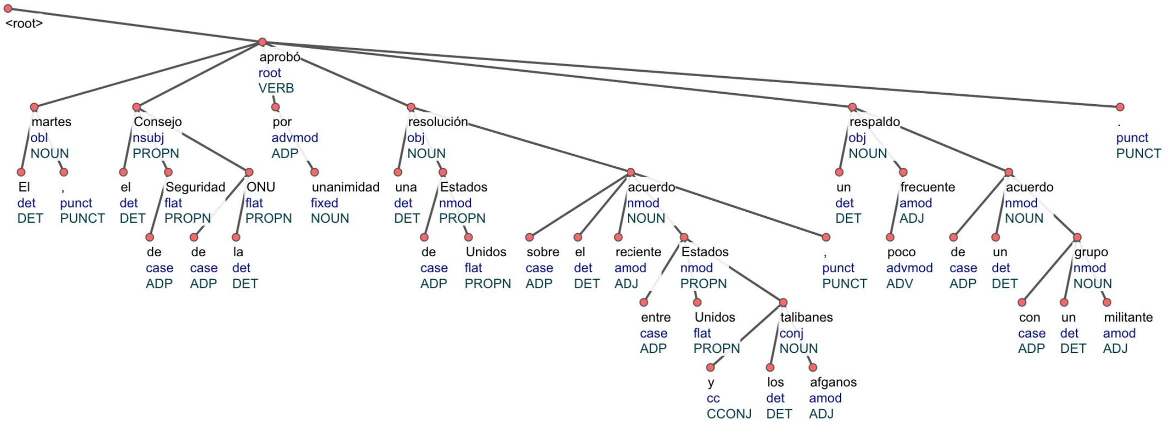


Fig. 5: Universal dependency tree for Sentence 1 in Spanish

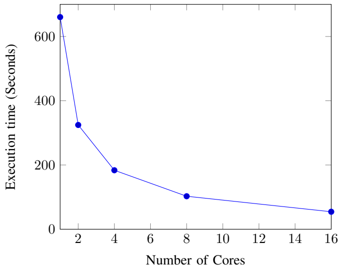


Fig. 6: Relation between execution time and number of available processing cores.

Classifier	Accuracy
Naive Bayes	81.4
Support Vector Machine (SBF)	80.7
Random Forest	**82.7**
BERT	74.6

TABLE IV: Accuracy of different classifiers

Ontology Translation

Here we discuss about the verb pattern translation app (VTA), where first the coders select correct English synonym sets from WordNet w.r.t a particular CAMEO code and then identify appropriate translation using Spanish synsets. They provided 10,358 correct verdicts (35.8% of the total verdicts) for English synonym set. Of those correct sysnsets, 90.1% of the Spanish translations are regarded as correct which shows the effectiveness of WordNet [15] based translation. The verb translation app [16] also provides valuable insights for inter-annotator reliability. The system tracks individual annotator's verdicts and generates an overall sense of agreement among them. In general, there is a correlation factor of 0.96 between the average proportion of correct and incorrect verdicts across users. This indicates a high degree of agreement between annotators in which they consistently identify English synsets as correct when they indeed correspond to the corresponding CAMEO concept, and consistently tag them as incorrect when their definition does not align with CAMEO's meaning. [16]

V. Related Works

In this section we present the related works with respect to the contents of the paper. Distributed processing of large data has been addressed by [26], [27] where author address a static dataset and the process of generating events. Here, we are working on a real-time dataset and here we need to collect and distribute the data in real-time where aforementioned works only concentrate on processing the data. We are inspired by the scalability analysis found here [26] and incorporate that to design the system. Schordt [28] has pointed out the importance of having a real-time event coding framework and data distribution. Automated Political event coding has gone through several decades and several ontologies, datasets and tools are developed. We use CAMEO ontology here. Other related ontologies are WEIS and authors [29] has provided a comparative study between those ontologies. Eck [30] also present an analysis among different conflict data-sets.

Among the available datasets ICEWS [31], Cline Center Dataset [32] are prominent. But they are not easily accessible and updated in-frequently. In terms of event coder, we found PETRARCH2 is the most flexible, easy-to-extend compared to others (i.e., BBN Accent) as reported by [33], [34]. We are also using a extended version of the event coder which uses Universal Dependency [7] to better support towards foreign languages. Such type of extension was not possible to BBN Accent like proprietary software.

Another dataset that matches with some types of events in CAMEO ontology would be the Global Terrorism Dataset(GTD) [35] which lists worldwide terrorist activities including different types of protest. The key aspect of the dataset is it is human annotated but doesn't link well with the original news source. We are currently working with this dataset to use it as a benchmark tool for the automatic event coder.

VI. Conclusion and Future Works

In this paper we described our infrastructure for real-time automated multilingual event coding in a distributed manner for better scalability. We presented some statistics and experimental results to show effectiveness of the individuls modules and system as a whole.

In the future, we will extend the system to capture other social and political phenomena. Currently we are working on a limited set of Spanish news sources. Once we increase that and add Arabic news sources, we need to study how the system performs in-terms of scalability. Moreover, we will assess the performance of the UD-PETRARCH event coder in other languages including Arabic, French and Portuguese. We will run our annotation process again for new articles to increase the accuracy of the DNN models used in the paper.

VII. Acknowledgments

The research reported herein was supported in part by NIH 1R21TW010991-01A1; NSA award H98230-15-1-0271; NSF FAIN awards DGE-1931800, OAC 1931541, and DGE-1723602; NSF awards DMS-1737978 and MRI-1828467. Any opinions, recommendations, or conclusions expressed are those of the authors and not necessarily of the aforementioned supporters.

References

[1] P. A. Schrodt, “Cameo: Conflict and mediation event observations event and actor codebook,” *Pennsylvania State University*, 2012.

[2] L. Khan and D. McLeod, “Effective retrieval of audio information from annotated text using ontologies,” in *Proceedings of the International Workshop on Multimedia Data Mining, MDM/KDD’2000, August 20th, 2000, Boston, MA, USA*, 2000, pp. 37–45.

[3] S. Abrol and L. Khan, “Twinner: understanding news queries with geo-content using twitter,” in *Proceedings of the 6th Workshop on Geographic information Retrieval*, 2010, pp. 1–8.

[4] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, “The stanford corenlp natural language processing toolkit,” in *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, 2014, pp. 55–60.

[5] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, and I. Stoica, “Apache spark: A unified engine for big data processing,” *Commun. ACM*, vol. 59, no. 11, pp. 56–65, Oct. 2016. [Online]. Available: http://doi.acm.org/10.1145/2934664

[6] J. Kreps, N. Narkhede, J. Rao *et al.*, “Kafka: A distributed messaging system for log processing,” 2011.

[7] R. McDonald, J. Nivre, Y. Quirmbach-Brundage, Y. Goldberg, D. Das, K. Ganchev, K. Hall, S. Petrov, H. Zhang, O. Täckström *et al.*, “Universal dependency annotation for multilingual parsing,” in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, vol. 2, 2013, pp. 92–97.

[8] M. Straka and J. Straková, “Tokenizing, pos tagging, lemmatizing and parsing ud 2.0 with udpipe,” pp. 88–99, August 2017. [Online]. Available: http://www.aclweb.org/anthology/K/K17/K17-3009.pdf

[9] I.-L. Yen, J. Goluguri, F. Bastani, L. Khan, and J. Linn, “A component-based approach for embedded software development,” in *Proceedings Fifth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing. ISIRC 2002*. IEEE, 2002, pp. 402–410.

[10] OEDA. (2018) List of news sources in spanish. [Online]. Available: https://docs.google.com/spreadsheets/d/13DmJ140wW8pCp6nyRSAk911S7AoF-6zJOJ-F77qoMuM/

[11] A. Haque, L. Khan, M. Baron, B. Thuraisingham, and C. Aggarwal, “Efficient handling of concept drift and concept evolution over stream data,” in *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, 2016, pp. 481–492.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[13] J. Lu. (2019) Universal dependency based petrarch, language-agnostic political event coding using universal dependencies. [Online]. Available: https://github.com/openeventdata/UniversalPetrarch

[14] C. D’Ignazio, R. Bhargava, E. Zuckerman, and L. Beck, “Cliff-clavin: Determining geographic focus for news,” *NewsKDD: Data Science for News Publishing, at KDD*, vol. 2014, 2014.

[15] C. Fellbaum, *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

[16] J. Osorio, V. Pavon, S. Salam, J. Holmes, P. T. Brandt, and L. Khan, “Translating cameo verbs for automated coding of event data,” *International Interactions*, vol. 45, no. 6, pp. 1049–1064, 2019.

[17] M. A. Helou, M. Palmonari, and M. Jarrar, “Effectiveness of automatic translations for cross-lingual ontology mapping,” *Journal of Artificial Intelligence Research*, vol. 55, pp. 165–208, 2016.

[18] R. Navigli and S. P. Ponzetto, “BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network,” *Artificial Intelligence*, vol. 193, pp. 217–250, 2012.

[19] F. P. Miller, A. F. Vandome, and J. McBrewster, *Levenshtein Distance: Information Theory, Computer Science, String (Computer Science), String Metric, Damerau?Levenshtein Distance, Spell Checker, Hamming Distance*. Alpha Press, 2009.

[20] D. Müllner, “Modern hierarchical, agglomerative clustering algorithms,” *arXiv preprint arXiv:1109.2378*, 2011.

[21] L. Wang, L. Liu, and L. Khan, “Automatic image annotation and retrieval using subspace clustering algorithm,” in *Proceedings of the 2nd ACM international workshop on Multimedia databases*, 2004, pp. 100–108.

[22] Wikipedia contributors, “Elbow method (clustering) — Wikipedia, the free encyclopedia,” 2019, [Online; accessed 20-August-2019]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Elbow_method_(clustering)&oldid=910764678

[23] H. Kim, V. D’Orazio, P. Brandt, J. Looper, S. Salam, L. Khan, and M. Shoemate, “Utdeventdata: An r package to access political event data,” *Journal of Open Source Software*, vol. 4, no. 36, p. 1322, 2019.

[24] Real-time event data server. [Online]. Available: https://github.com/Sayeedsalam/spec-event-data-server

[25] (2020) Google translation pricing. [Online]. Available: https://cloud.google.com/translate/pricing

[26] M. Solaimani, R. Gopalan, L. Khan, P. T. Brandt, and B. Thuraisingham, “Spark-based political event coding,” in *2016 IEEE Second International Conference on Big Data Computing Service and Applications (BigDataService)*, 2016, pp. 14–23.

[27] A. Halterman, J. Irvine, M. Landis, P. Jalla, Y. Liang, C. Grant, and M. Solaimani, “Adaptive scalable pipelines for political event data generation,” in *2017 IEEE International Conference on Big Data (Big Data)*, Dec 2017, pp. 2879–2883.

[28] P. A. Schrodt, “Precedents, Progress, and Prospects in Political Event Data Article in International Interactions,” 2012. [Online]. Available: https://www.researchgate.net/publication/254242323

[29] D. J. Gerner, P. A. Schrodt, O. Yilmaz, and R. Abu-Jabr, “Conflict and mediation event observations (cameo): A new event data framework for the analysis of foreign policy interactions,” *International Studies Association, New Orleans*, 2002.

[30] K. Eck, “In data we trust? a comparison of ucdp ged and acled conflict events datasets,” *Cooperation and Conflict*, vol. 47, no. 1, pp. 124–141, 2012.

[31] E. Boschee, J. Lautenschlager, S. O’Brien, S. Shellman, J. Starz, and M. Ward, “ICEWS Coded Event Data,” 2018. [Online]. Available: https://doi.org/10.7910/DVN/28075

[32] J. B. J. F. C. B. P. Althaus, Scott and D. A. Shalmon, “Cline Center Historical Phoenix Event Data. v.1.0.0.” 2017. [Online]. Available: http://www.clinecenter.illinois.edu/data/event/phoenix/

[33] J. Beieler, “Generating politically-relevant event data,” *arXiv preprint arXiv:1609.06239*, 2016.

[34] M. Solaimani, S. Salam, L. Khan, P. T. Brandt, and V. D’Orazio, “Repair: Recommend political actors in real-time from news websites,” in *2017 IEEE International Conference on Big Data (Big Data)*, Dec 2017, pp. 1333–1340.

[35] Global terrorism dataset. [Online]. Available: https://www.start.umd.edu/gtd/