Proceedings of AESPEN 2020, pages 19–25
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020

Text Categorization for Conflict Event Annotation

Fredrik Olsson, Magnus Sahlgren, Fehmi Ben Abdesslem, Ariel Ekgren, Kristine Eck
RISE. Uppsala University
Intelligent Systems. Department of Peace and Conflict Research
Box 1263, 164 29 Kista, Sweden. Box 514, 751 20 Uppsala, Sweden
{fredrik.olsson, magnus.sahlgren, fehmi.ben.abdesslem, ariel.ekgren}@ri.se, kristine.eck@pcr.uu.se

Abstract

We cast the problem of event annotation as one of text categorization, and compare state of the art text categorization techniques on event data produced within the Uppsala Conflict Data Program (UCDP). Annotating a single text involves assigning the labels pertaining to at least 17 distinct categorization tasks, e.g., who were the attacking organization, who was attacked, and where did the event take place. The text categorization techniques under scrutiny are a classical Bag-of-Words approach; character-based contextualized embeddings produced by ELMo; embeddings produced by the BERT base model, and a version of BERT base fine-tuned on UCDP data; and a pre-trained and fine-tuned classifier based on ULMFiT. The categorization tasks are very diverse in terms of the number of classes to predict as well as the skewness of the distribution of classes. The categorization results exhibit a large variability across tasks, ranging from 30.3% to 99.8% F1-score.

Keywords: Event detection, Text categorization, Language models

1. Introduction

This study concerns the application of automatic text categorization techniques for the purpose of conflict event annotation using the data of the Uppsala Conflict Data Program.[1] In the terminology of UCDP, an event is an instance of fatal organized violence, defined by Sundberg and Melander (2013) as:

> The incidence of the use of armed force by an organized actor against another organized actor, or against civilians, resulting in at least 1 direct death in either the best, low or high estimate categories at a specific location and for a specific temporal duration

The present study seeks to investigate the automation of event annotation by taking advantage of recent advances in representation and transfer learning to harness the power of pre-trained and fine-tuned language models for representing the textual data subject to categorization. The purpose is to assess the relative performance of text categorization when the learner has access to language knowledge beyond that which is present in the training corpus, across a multitude of categorization tasks.

2. Related work

Document categorization, or document classification, consists in assigning one or several pre-defined labels, based on the contents of a whole document (here, a news article). In its simplest form, document categorization does not require that the ordering of tokens (or even the structures in which the tokens are arranged) is retained while extracting information. To the best of our knowledge, such document categorization introduced in this paper has not previously been applied to news articles for the purpose of event coding. Instead, however, sequence classification has been the focus of several works to automate the event encoding from news articles. Sequence classification is first based on the extraction of information, that is then used for attributing the characteristics of an event (such as the dyad[2] or the number of deaths) described in a document. Information extraction is typically based on classification tasks in which each unit (character, character sequence or token) in a text is classified as to whether it refers to a named entity (actors, location), time, number of casualties, or any other event characteristics.

In particular, there are several projects aiming at automating political event coding with sequence classification. The KEDS (Kansas Event Data System) project (Schrodt et al., 1994) was one of the first attempts, and was mainly based on parsing text to extract words that are pre-defined in dictionaries (actors and verbs).

TABARI (Schrodt, 2009) replaced KEDS by introducing significant improvements such as recognizing passive-voice sentences or disambiguating verbs that can also be nouns (e.g., Attack). TABARI was then replaced by Petrarch (Norris et al., 2017) and Universal Petrarch.

Petrarch stands for "Python Engine for Text Resolution And Related Coding Hierarchy". As its aforementioned predecessors, it is also a processing tool for machine-coding text describing events (i.e. news articles). It is designed to process fully-parsed news summaries, from which "whom-did-what-to-whom" relations are extracted. The output is then a dyad and an action. Date and location are also extracted. Petrarch is typically used by running the Phoenix pipeline,[3] which mainly consists in the following steps:

1. Extract articles and corresponding date from online sources using a web scraper[4].
2. Encode the sentences with Named Entity Recognition (NER) using Stanford CoreNLP (Manning et al.,

[1] `https://ucdp.uu.se`

[2] "A dyad is made up of two armed and opposing actors." See: `https://www.pcr.uu.se/research/ucdp/definitions/`

[3] `https://phoenix-pipeline.readthedocs.io/`

[4] `https://github.com/openeventdata/scraper`

2014)

3. Encode each sentence with [source_actor, action, and target_actor] (who does what to whom) using Petrarch.
4. Encode each sentence with a location using CLIFF-CALVIN (D'Ignazio et al., 2014) or Mordecai (Halterman, 2017).

In all these tools, actors and actions (verbs) are pre-defined in a specific ontology. Both Petrarch and Universal Petrarch use the same ontology for actors and verbs, based on TABARI dictionaries. TABARI dictionaries follow the CAMEO (Conflict and Mediation Event Observations) framework (Schrodt et al., 2008), which was initially intended as an extension of an ontology from the 60-70s called WEIS (McClelland, 2006). Another old ontology is COPDAB (Azar, 1980) in the 1980s. Competing modern ontologies to CAMEO are the IDEA (Bond et al., 2003) ontology from the 2000s, and the JRC-names (Ehrmann et al., 2017) in the 2010s, developed as a by-product of the EMM (European Media Monitor) project.
Currently, CAMEO is being replaced by PLOVER,[5] a new ontology with coverage of some new actions, vastly simplified coding of other actions, and a more flexible system for extensions and modifications.
Coding systems such as Petrarch and Universal Petrarch are rule-based: they use rules to decide which noun phrases are actors and which verb phrases are actions, and then compare these chunks of text against lists of hand-defined rules for coding actions and actors. Despite using NLP methods (e.g., NER), they are rarely using advanced machine learning algorithms. Among the few works using machine learning we can cite the work of Beieler (2016), who uses a character-based convolutional neural network, based on the work of Zhang et al. (2015), to determine the type of event action. However, the event actors are still determined with Petrarch, and the training dataset is also labelled with Petrarch.
Recently, categorizing news articles has also been experimented by Adhikari et al. (Adhikari et al., 2019) using BERT (introduced in Section 5.4.) to extract the topic of the articles.

3. Event annotation at UCDP

The Uppsala Conflict Data Program is the oldest ongoing data collection project for civil war, dating back almost 40 years. UCDP continuously updates its online database on armed conflicts and organized violence, in which information on several aspects of armed conflict such as conflict dynamics and conflict resolution is available. The database offers a web-based system for visualizing, handling and downloading data, including ready-made datasets on organized violence and peacemaking, all free of charge. UCDP is staffed by permanent full-time employees, handling data collection and processing detailed in (Högblad, 2019), including analysis and management.
The typical work-flow for a UCDP event annotator amounts to the following. For retrieving the news data from their data provider, an annotator:

[5]`https://github.com/openeventdata/PLOVER`

1. inputs search terms to search selected news sources, then;
2. judges whether each news item retrieved:
   (a) describes a conflict event relevant to UCDP, and
   (b) either describes a new event, or brings new information about a known event.

Once a news text passes the above criteria, i.e., it is in fact relevant and contributes new information, the annotator looks for the following information in it:

- Geography (country, region, and even finer grained geographical reference points).
- Participants in the dyad.
- The number of deaths reported.
- Date or time period of the event.

More often than not, multiple news items relating to the same event are required in order to decide on all of the aforementioned attributes for an event. UCDP staff processes approximately 50 000 news items and other reports yearly, depending on the conflict situation in the world. In total, each text is manually annotated with up to 19 different labels.
The textual data in the UDCP database is annotated at the document level, rather than with in-text annotations at the sentence level. For instance, a document annotated with information about the dyad being part of an event exhibits an association between the dyad identifier and the document, but it does not provide information as to where in the document the reference to the dyad is located, and thus not how the surface form of the reference is manifested. This is a consequence of how the UCDP staff work when annotating event data, and it renders it natural to cast the event annotation problem as one of text categorization, rather than as a sequence extraction and labelling task. The annotation tasks consist in identifying the labels present in Table 1.

4. The dataset

The dataset at hand in this study consists of a combination of two distinct sources; the internal UCDP database compiled while UCDP annotators are working with identifying events in news text and reports, and the externally published Georeferenced Event Dataset (Sundberg and Melander, 2013). The former contains textual information related to the source documents read by the annotator while annotating the event, while the externally published event data is a clean, quantitative view of the text data. The combination of the data sources constitutes the ground truth, that the machine learning experiments carried out in this study will try to re-create.

4.1. The training set

The dataset used in the following experimental setup consists of 31 772 UCDP events, each of which is associated with a unique body of text in English. A body of text can consist of a (mix of) notes made by the annotator, records

Label	Description	Number of classes	Class entropy
side_a	Name of state or government side involved.	299	3.9
side_b	Name of other participant.	301	3.5
dyad_name	Combination of side_a and side_b.	510	4.5
type_of_violence	State-based, non-state, or one sided.	3	0.8
conflict_name	The name of the conflict.	428	4.3
where_coordinates	Name of place of conflict.	4 125	7.4
region	Name of region.	5	1.4
country	Name of country.	84	3.2
adm_1	More precise name of region.	672	5.3
adm_2	Even more precise name of region.	1 739	6.5
deaths_a	Number of deaths reported for side_a.	75	1.4
deaths_b	Number of deaths reported for side_b.	115	1.9
deaths_civilians	Number of deaths reported for civilians.	117	1.5
deaths_unknown	Number of deaths reported for unknown side.	104	0.9
low	The lowest estimate of number of deaths reported for event.	175	3.2
best	The best estimate of number of deaths reported for event.	187	3.2
high	The highest estimate of number of deaths reported for event.	218	3.3

Table 1: The labels to be identified by tasks, along with their short descriptions, their number of classes, and their class entropy for the dataset consisting of 31 772 events. The class entropy is a measure of the class imbalance for a task such that a low value indicate higher imbalance. The class entropy is elaborated on in Section 4.2..

copied verbatim from an online conflict tracker, part or the whole of one or several news items, or some other distinct unit of text taken from an online resource. The dataset has been pre-processed and chosen so as to make sure that each text has given rise to a unique UCDP event. That is, in the current dataset there is a one-to-one relation between a body of text and an event. Thus, all texts that have resulted in two or more UCDP events have been omitted. The rationale behind this decision is the following: if a machine cannot reproduce the accuracy of the human annotators when presented with an admittedly simplified scenario (i.e., expect no more than exactly one event per text), then it will not perform well in a more realistic setting either (i.e., expect an arbitrary number of events to be described in each text). Only if the results in the simpler scenario are satisfactory should the more complicated setting be addressed.

4.2. The labels to predict

There are at least 17 different categorization tasks that a UCDP annotator has to deal with for every single event (omitting the temporal categories, i.e., the starting and ending date of an event). The annotations of the event data provided by UCDP constitutes the ground truth, and is as such the target of the predictions in the experiments to follow. In other words, for each of the bodies of texts in the dataset, there are 17 labels to predict. Table 1 shows the possible number of different classes that are in play in each of the annotation tasks, as well as the normalized entropy among those labels. The normalized class entropy value η is defined as $\eta(X) = -\sum_{i=1}^{n} \frac{p(x_i)\ln(p(x_i))}{\ln(n)}$ where X is the set of n possible classes, and $p(x_i)$ is the observed fraction of values equal to the ith class. The entropy is indicative of the distribution of classes within a task. A low entropy value is a sign of a skewed distribution, e.g., one class is significantly more frequent than the others, while a high entropy implies a more even distribution of classes. Combined, the size of the data, the number of classes and the class entropy tells us something about the expected complexity of the annotation task. For example, given the values in Table 1, it is expected that the task where_coordinates will be hard since it contains many classes (4 125) that are relatively evenly distributed across the dataset (the entropy value is high) giving, on average, relatively few events per class (31 722/4 125) to learn from. On the other hand, the task type_of_violence task exhibits a number of classes and class entropy at the other end of the spectrum: it is comprised of few classes (3) that are unevenly distributed in terms of occurrences in the dataset (entropy 0.8). Thus, an annotator is expected to perform well for (the majority) classes in the task.
Of course, there is more than meets the eye when it comes to how well a classifier actually manages to perform than just the number of classes, and their relative distribution, but these numbers give a hint as to what to expect.

5. Experimental setup

The experiments carried out in this study involve learning from the contents of the texts described in Section 4.1. to predict the classes of each task described in Section 4.2.. There are 17 different tasks, each of which will be addressed using five different text categorization techniques, as well as a random guessing-based baseline performance estimation.
For each task, the baseline (Section 5.1.), Bag-of-Words (BoW, Section 5.2.), ELMo experiments (Section 5.3.), the two BERT versions (Section 5.4.) are based on 5-fold cross-validation, with test data size set to 20% of the total corpus. This means that the baseline, BoW, ELMo, and BERT results are supported by approximately 30 000 data points each. Due to the time it took to complete the ULMFiT experiments (Section 5.5.), they are based on a single training and testing set, where the testing set is made

up of approximately 6 000 data points, instead of the 5-fold cross-validation scheme employed in the other experiments. The split into training and testing data used by ULMFiT corresponds to the first fold in the baseline, BoW, ELMo and BERT cases, as it is made with the same logic and settings.

5.1. Baseline

A "dummy" classifier that guesses the class of a text by randomly drawing a class label from the class label distribution is used to assess a baseline upon which the machine learning-based classifiers should improve. The dummy classifier is available in scikit learn described by Pedregosa et al. (2011).

5.2. Using a standard Bag-of-Words approach

A classical way to represent documents in text categorization is as a collection of words, in which the order of the words is assumed to be irrelevant. This type of representation is usually referred to as Bag-of-Words. The assumption is naïve, but historically, it has produced relatively competitive results. The BoW representation used in the current setup contains single words (unigrams), as well as all combinations of two consecutive words in the training corpus (bigrams). A linear learning method (Logistic Regression) is then used to train classifiers to distinguish between the classes in the different tasks.
The BoW approach is included in the experiments since it, in the past, has been a go-to solution in many text categorization tasks and thus constitutes a sensible baseline that more modern approaches should beat.

5.3. ELMo

Embeddings from Language Models (ELMo) described by (Peters et al., 2018), is a deep character-based neural network that learns embeddings by predicting the next token given an input sequence. The network architecture includes both convolutional and (bidirectional) LSTM layers, and produces an embedding that that is sensitive to the particular context of the input sequence. Contextualized embeddings have proven to be highly beneficial when using the embeddings as representation in downstream natural language processing tasks such as categorization, entity recognition, and question answering. In the current setup, an existing pretrained version[6] of ELMo is used to produce a single 1 024 elements long feature vector for the body of text associated to each event in the UCDP data. The data used for pretraining the ELMo model used here is reported to be approximately 20 million randomly selected texts from Wikipedia and CommonCrawl, amounting to a total training time of 3 days per language. The ELMo feature vectors are then used as input to a non-linear learner (Random Forest) to train a classifier for distinguishing between the classes in each of the 17 tasks.
The ELMo approach is included in the experiments since it has proven to be a simple and effective way of incorporating language knowledge in machine learning situations where training data is scarce.

[6]`https://github.com/HIT-SCIR/ELMoForManyLangs`

5.4. BERT

Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) is a deep, attention-based neural network architecture that produces a contextualized representation of a text by taking both the left and right context into account simultaneously. In this respect, it differs from ELMo, which builds its representation of text based on a concatenating representations from the left and right context. Since its inception, BERT has been shown to improve the state-of-the art on many language processing tasks, including some text categorization ones.
In the experiments to follow, we use two versions of BERT: the original large pre-trained uncased base model made available via Hugging Face's Transformers (Wolf et al., 2019), and a version of the same model fine-tuned on the UCDP data.

5.5. ULMFiT

Universal Language Modelling Fine-Tuning (ULMFiT), described in (Howard and Ruder, 2018), is a three step method for transferring general language use to specific categorization tasks. The method consists of the following three steps:

1. Train a language model on an unannotated corpus of general language.
2. Fine-tune the language model based on unannotated in-domain texts.
3. Train and fine-tune a text classifier on annotated texts.

An initial language model (Step 1) is readily available online. ULMFiT is pretrained on a subset of the English Wikipedia containing more than 103 million running words taken from more than 28 000 verified *Good* or *Featured* articles (Merity et al., 2016). In Step 2, we used the texts associated with the 31 722 UCDP events to fine-tune the language model. Finally, in Step 3, a classifier was created for each of the 17 different tasks outlined in Table 1.
The implementation of ULMFiT used in the current experiment is based on the AWD-LSTM language model architecture described by (Merity et al., 2017).
The ULMFiT approach is included in the experiments because it is a robust method for leveraging the language knowledge of a pretrained model and its ability to adjust that model based on in-domain data, without requiring vast computational resources. Until recently, ULMFiT produced state-of-the art classifiers for a number of benchmarks.

6. Categorization results

Table 2 on the next page shows the results from the experiments in terms F1-score for the random baseline, the BoW-based Logistic Regression classifier, the ELMo-based Random Forest classifier, the original and fine-tuned BERT-based Random Forest classifiers, as well as for ULMFiT. As an example, refer back to the discussion of the complexity of the annotation tasks in terms of the number of classes and the class entropy in Section 4.2., and consider the baseline F1-score result for the task type_of_violence

Table 2: UCDP document categorization results.

Task	Cls	En	B.F	BW.F	E.F	BE.F	BF.F	U.F
side_a	299	3.9	5.0	76.8	76.2	81.1	**84.9**	84.7
side_b	301	3.5	8.1	73.7	75.5	78.3	82.0	**82.5**
dyad_name	510	4.5	4.1	66.9	72.5	75.6	79.3	**80.8**
type_of_violence	3	0.8	56.6	88.8	85.8	88.6	89.6	**91.8**
conflict_name	428	4.3	4.2	69.5	73.4	76.9	80.7	**82.7**
where_coordinates	4125	7.4	0.3	▨	▨	▨	▨	**30.3**
region	5	1.4	28.7	99.4	89.6	97.7	98.7	**99.8**
country	84	3.2	6.9	95.5	82.8	90.2	94.7	**97.4**
adm_1	672	5.3	1.0	64.2	62.2	62.8	65.1	**77.7**
adm_2	1739	6.5	0.4	27.5	▨	▨	▨	**41.3**
deaths_a	75	1.4	46.8	63.6	**83.1**	82.2	82.2	73.3
deaths_b	115	1.9	35.6	59.0	75.1	74.8	**75.5**	67.4
deaths_civilians	117	1.5	48.7	63.8	**84.1**	83.5	83.7	70.9
deaths_unknown	104	0.9	72.5	79.0	**93.3**	92.7	92.7	80.8
low	175	3.2	8.5	32.3	**61.6**	58.5	58.5	37.9
best	187	3.2	8.3	32.6	**61.1**	58.1	58.4	41.6
high	218	3.3	8.5	32.4	**61.8**	58.6	58.7	40.0

Task	The name of the annotation task.
Cls	The number of distinct classes for a particular task.
En	The class entropy: a high value corresponds to a more evenly distribution of instances per class.
B	Baseline, random guessing based on distribution of labels.
BW	Bag of words representation.
E	ELMo representations + non-linear classifier.
BE	BERT representations + non-linear classifier.
BF	BERT representations, model fine-tuned on UCDP data + non-linear classifier.
U	ULMFiT pretrained on Wikipedia, fine-tuned and trained on UCDP data.
F	weighted F1-score.
▨	Light grey cells in the table indicate a failure of the classifier to complete the corresponding task. The failures are due to the size of the models: for tasks with many classes, the memory consumption of the learner exceeds that of the available memory (which in this case is 255Gb).

which is given in column B.F in Table 2. The task concerns only three highly imbalanced classes, which in effect means it is easy to get a fairly good score just by making a vaguely informed guess with respect to the class. The random guessing-based baseline F1-score is 56.6%. All trained classifiers improve on the baseline, with ULMFiT performing the best at an F1-score of 91.8%, a 35.2 percent point improvement.

The other example in Section 4.2. is that of where_coordinates. The baseline results for the task align with the expected outcome given the size of the data, the number of classes, and the class entropy: the F1-score value is low, at around 0.3% of a possible 100%. The ULMFiT classifier improves the F1-score given the baseline with 30.0%. Still, at an F1-score of 30.3%, the classifier clearly underperforms vis-à-vis the human annotated data.

According to Table 2, the tasks that the hardest for the classifiers are:

- where_coordinates (ULMFiT F1-score: 30.3%)
- adm_2 (ULMFiT F1-score: 41.3%)
- low (ELMo F1-score: 61.6%)
- best (ELMo F1-score: 61.1%)
- high (ELMo F1-score: 61.8%)

The above are all tasks in which there are many classes, and thus little data to learn from per class. The following are the tasks on which the classifiers performed the best:

- region (ULMFiT F1-score: 99.8%)
- country (ULMFiT F1-score: 97.4%)
- deaths_unknown (ELMo F1-score: 93.3%)
- type_of_violence (ULMFiT F1-score: 91.8%)
- side_a (BERT fine-tuned F1-score: 84.9%)
- deaths_civilians (ELMo F1-score: 84.1%)
- deaths_a (ELMo F1-score: 83.1%)
- conflict_name (ULMFiT F1-score: 82.7%)
- side_b (ULMFiT F1-score: 82.5%)
- dyad_name (ULMFiT F1-score: 80.8%)

However, it should be emphasized that the experimental setting in this report is a simplified one that only includes data in which each textual body corresponds to exactly one UCDP event.

7. Discussion

From the results of this study, we make two observations. The first observation concerns text categorization for event annotation, while the other is about the developments in the field of transfer learning in NLP.

7.1. Text categorization for event annotation

By casting the event annotation problem as one of text categorization, we have gained initial insight into the complexity of assigning values to the individual attributes of events. Some attributes are naturally harder to automatically predict than others: for instance, the finer-grained geographical location of an event (where_coordinates) is harder to assess than the immediately broader region (country). Similarly, the dyad name is harder to predict than the names of its participants. It is also clear that automated text categorization has value in that it performs very near the level of human annotators, for some tasks. This begs the question: How can we best make use of text categorization for the purpose of improving the human annotation process in terms of, e.g., speed, and consistency? We believe that the categorization results reported in this study are encouraging enough to warrant continued investigations with respect to its use in the manual annotation process, as well as further improvements of the categorization results. As for the latter, there are two immediate issues that require attention. The first issue is to go from the simplified setting of the current experiments to one that allows the more natural many-to-many relationship between texts and events. The second issue is to investigate methods for making use of the conditional dependencies between tasks e.g., that certain dyads are active only in certain geographical locations.

7.2. Transfer learning in NLP

Although the bag-of-words approach is a strong baseline, it is almost always better to utilize pre-training and fine-tuning on domain-specific data. ELMo and the original BERT model are both pre-trained on large amounts of data, and do not make use of any in-domain data in the current setting. Still, both models perform well, beating the BoW baseline in most cases. Furthermore, fine-tuning pre-trained models on domain-specific data always helps: the fine-tuned BERT model beats the original model across all tasks.

Acknowledgements

The study presented in this paper is funded by Riksbankens Jubileumsfond, via the research project *Automation of the Uppsala Conflict Data Program (UCDP)*, reference number IN18-0710:1.
The authors wish to thank the anonymous reviewers for valuable and thoughtful comments.

8. Bibliographical References

Adhikari, A., Ram, A., Tang, R., and Lin, J. (2019). Docbert: Bert for document classification. *ArXiv*, abs/1904.08398.

Azar, E. E. (1980). The conflict and peace data bank (COPDAB) project. *Journal of Conflict Resolution*, 24(1):143–152.

Beieler, J. (2016). Generating politically-relevant event data. *arXiv preprint arXiv:1609.06239*.

Bond, D., Bond, J., Oh, C., Jenkins, J. C., and Taylor, C. L. (2003). Integrated data for events analysis (IDEA): An event typology for automated events data development. *Journal of Peace Research*, 40(6):733–745.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

D'Ignazio, C., Bhargava, R., Zuckerman, E., and Beck, L. (2014). Cliff-clavin: Determining geographic focus for news. *News KDD: Data Science for News Publishing*, 2014.

Ehrmann, M., Jacquet, G., and Steinberger, R. (2017). Jrc-names: Multilingual entity name variants and titles as linked data. *Semantic Web*, 8(2):283–295.

Halterman, A. (2017). Mordecai: Full text geoparsing and event geocoding. *The Journal of Open Source Software*, 2(9).

Högblad, S. (2019). UCDP GED Codebook version 19.1.

Howard, J. and Ruder, S. (2018). Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., and McClosky, D. (2014). The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

McClelland, C. (2006). World Event/Interaction Survey (WEIS) Project, 1966-1978.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. (2016). Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Merity, S., Keskar, N. S., and Socher, R. (2017). Regularizing and optimizing lstm language models. *arXiv preprint arXiv:1708.02182*.

Norris, C., Schrodt, P., and Beieler, J. (2017). PETRARCH2: Another event coding program. *The Journal of Open Source Software*, 2(9), 1.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Schrodt, P. A., Davis, S. G., and Weddle, J. L. (1994). Political Science: KEDS - A Program for the Machine Coding of Event Data. *Social Science Computer Review*, 12(4):561–587.

Schrodt, P. A., Yilmaz, O., Gerner, D. J., and Hermreck, D. (2008). The CAMEO (conflict and mediation event observations) actor coding framework. In *2008 Annual Meeting of the International Studies Association*.

Schrodt, P. A. (2009). Tabari: Textual analysis by augmented replacement instructions. *Dept. of Political Science, University of Kansas, Blake Hall, Version 0.7. 3B3*, pages 1–137.

Sundberg, R. and Melander, E. (2013). Introducing the ucdp georeferenced event dataset. *Journal of Peace Research*, 50(4):523–532.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J. (2019). Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.