Extracting Relations from Italian Wikipedia Using Self-Training

Lucia Siciliani[1], **Pierluigi Cassotti**[1], **Pierpaolo Basile**[1], **Marco de Gemmis**[1], **Pasquale Lops**[1] **and Giovanni Semeraro**[1]
[1] Dipartimento di Informatica
Università degli Studi di Bari Aldo Moro, Bari, Italy
{firstname}.{surname}@uniba.it

Abstract

In this paper, we describe a supervised approach for extracting relations from Wikipedia. In particular, we exploit a self-training strategy for enriching a small number of manually labeled triples with new self-labeled examples. We integrate the supervised stage in WikiOIE, an existing framework for unsupervised extraction of relations from Wikipedia. We rely on WikiOIE and its unsupervised pipeline for extracting the initial set of unlabelled triples. An evaluation involving different algorithms and parameters proves that self-training helps to improve performance. Finally, we provide a dataset of about three million triples extracted from the Italian version of Wikipedia and perform a preliminary evaluation conducted on a sample dataset, obtaining promising results.

1 Introduction

The goal of an Open Information Extraction (Open IE) system is to extract relations occurring within a text written in natural language. Each relation is structured in the form of a triple that is composed by three elements i.e. $\{(arg1; rel; arg2)\}$. More specifically, given a relation, $arg1$ and $arg2$ can be nouns or phrases, while rel is a phrase that denotes the semantic relation between them. Open IE finds its application in several NLP tasks like Question Answering, Knowledge Graph Acquisition, Knowledge Graph Completion, and Text Summarization. For this reason, Open IE is gaining ever-growing attention as a research topic. Given the nature of the task, approaches for Open IE are deeply intertwined with the language of the corpora that have to be analyzed. Due to the availability of English corpora, the majority of the state-of-the-art works are specific for that language. For what concerns the Italian language, the model proposed by Guarasci et al. (2020) relies on verbal behavior patterns based upon Lexicon-Grammar features. In a previous work, we proposed WikiOIE (Cassotti et al., 2021), a framework in which Open IE methods for the Italian language can be easily developed with the aim of encouraging researchers to conduct further work also for under-represented languages. The first solutions developed in WikiOIE are unsupervised, relying merely on PoS tags patterns and dependency relations. In Cassotti et al. (2021) the triples extracted by WikiOIE underwent a deep error analysis. The error analysis reveals syntactic errors such as missing subject or incomplete object information and semantic errors such as generic subject or relation. In this work, we propose a supervised approach to automatically filter out non-relevant triples provided by WikiOIE and a self-training strategy. Self-training (Yarowsky, 1995) works iteratively: a classification model is trained on labeled data, the trained model is used to classify unlabeled data i.e. pseudo-labels, the classification model is retrained on labeled data and high-confident pseudo-labels. Specifically, we manually annotate a small number of triples extracted by WikiOIE. Afterward, the annotated triples are augmented using self-training. Finally, the set of triples obtained through self-training at the previous step is exploited to train a supervised model. The paper is structured as follows: after a brief introduction of state-of-the-art methods for Open IE, Section 3 provides details about the self-training and the supervised model behind our methodology. Section 4 reports the results of the evaluation, while Section 5 closes the paper.

2 Related Work

At first, the IE task was performed by extracting from the text relations that were defined a-priori. However, the increasing amount of corpora available nowadays makes this process unfeasible, thus creating the urge to propose novel solutions to tackle this problem.

The Open IE task was defined in 2008 by Etzioni et al. (2008). The three most important elements characterizing this task are the following: it is domain independent, meaning that the text relations must be extracted from, can be related to any topic, the extraction must be unsupervised, approaches to solve this task must take into account the amount of data available and must be scalable.

Along with the definition of a new task, the authors proposed a model called TextRunner. It applies an approach that is composed of three main modules. The first one is a *learner* that exploits a parser to label the training data as trustworthy or not and then uses the extracted information to train a Naive Bayes classifier. Next, the *extractor* uses POS-tag features to obtain a set of candidate tuples from the corpus, and only those labeled as trustworthy are kept. Finally, a module denominated *assessor* assigns a probability score to the tuples extracted at the previous step based on the number of occurrences in the corpus.

The learning-based approach used in TextRunner has also been applied by several other systems like WOE (Wu and Weld, 2010), OLLIE (Mausam et al., 2012), and ReNoun (Yahya et al., 2014). In particular, WOE exploits Wikipedia-based bootstrapping: the system extracts the sentences matching the attribute-value pairs available within the info-boxes of Wikipedia articles. This data is then used to build two versions of the system: the first one based on PoS-tags, regular expressions, and other shallow features of the sentence, the latter based on features of dependency-parse trees, thus obtaining better results than the other one but with a lack of performance in terms of speed.

In recent works, OIE has been treated as a sequence labeling task. In this setting, models are trained to extract triple elements, i.e., subject, predicate, and object using a modified BIO tag schema (Ratinov and Roth, 2009) that involves particular prefixes to represent the triple elements, i.e., A0, P, and A1. Hohenecker et al. (2020) provide an evaluation of different training strategies and different neural network architectures such as bidirectional Long short-term Memory (BiLSTM), Convolutional Neural Networks (CNNs), and Transformers improving the state-of-the-art on the OIE16 benchmark (Stanovsky and Dagan, 2016) which focuses on the English language.

3 Methodology

In this section, we describe our supervised approach based on self-training integrated into the information extraction system called WikiOIE[1]. Before discussing details about the supervised approach, it is necessary to recap how WikiOIE works. The input of the pipeline is represented by the textual format of the Wikipedia dump obtained through the WikiExtractor tool[2] (Attardi, 2015). The text is extracted from the Wikipedia dump and processed using the UDPipe tool (Straka and Straková, 2017). For this task, we use version 1 of UDPipe with version 2.5 of the *ISDT-Italian* model. We opt for UDPipe, since it is trained using Universal Dependencies data for over 100 languages. In this way, our system can be potentially used on different Wikipedia dumps of several languages. WikiOIE directly calls the REST API provided by UDPipe so that it is easy to change the endpoint and the model/language. Another advantage of using Universal Dependencies is the common tag-set that is defined for all the languages. PoS-tags[3] and syntactic dependencies[4] are annotated with shared sets of labels. Again, this feature also allows the system to be independent from the language. The Wikipedia dump is read line-by-line. Each line contains a fragment (passage) of text that is processed using UDPipe. The output of this process is a set of sentences, and each sentence is annotated with syntactic dependencies. The sentence is transformed into a dependency graph that is the input of the Wiki Extractor module. This module extracts facts from the sentence in the form of triples *(subject, predicate, object)* and assigns a score.

As aforementioned, each sentence occurring in the text is annotated by UDPipe that provides an-

[1]The code is available on GitHub: `https://github.com/pippokill/WikiOIE`.

[2]`https://github.com/attardi/wikiextractor/wiki/File-Format`

[3]`https://universaldependencies.org/u/pos/`

[4]`https://universaldependencies.org/u/dep/`

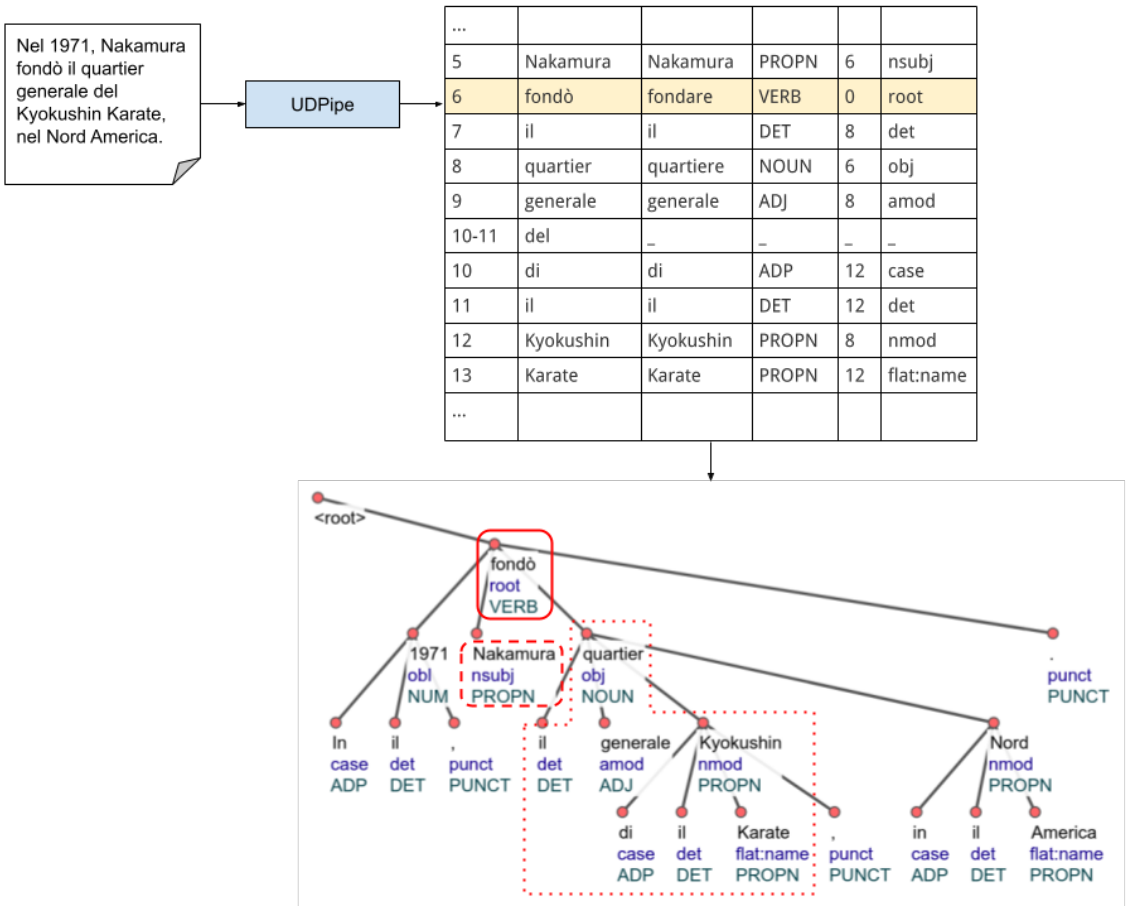

Figure 1: An example of UDPipe processing.

notations following the CoNLL-U format[5]. As shown in Figure 1, each token into the sentence is denoted by an index (first column) corresponding to the token position into the sentence (starting from 1). In the other columns are stored the features extracted by UDPipe, such as the token, the lemma, the universal PoS-tag, the head of the current word, and the universal dependency relation to the HEAD. If the head of the current word is equal to 0, it means that that token represents the head of the whole sentence, then the universal dependency relation will be equal to *root*. Figure 1 also reports the dependency graph of the sentence that is used by the Wiki Extractor module for extracting triples. We use an unsupervised pipeline based on both PoS-tag and dependencies to extract the first set of triples.

The first step of the extraction process consists of identifying sequences of PoS-tags that match verbs as reported in Table 1. In Table 1, the first column reports the PoS-tag patterns, while the second one reports an example of pattern usage. The sentence showed in Figure 1 matches the last pattern (VERB, *fondò*).

PoS-tag Pattern	**Example**
AUX VERB ADP	... *è nato nel* ...
AUX VERB	... *è nato* ...
AUX=(essere, to be)	... *è* ...
VERB ADP	... *nacque nel* ...
VERB	... *acquisì* ...

Table 1: Patterns of valid predicates.

When the information extraction algorithm finds a valid predicate pattern, it checks for a candidate subject and object for the predicate. A valid subject/object candidate must match the following constraints:

1. the candidate must be composed by a sequence of tokens belonging to the following PoS-tags: noun, adjective, number, determiner, adposition, proper noun;

2. the sequence of tokens composing the candi-

[5] https://universaldependencies.org/format.html

date can contain only one determiner and/or one adposition.

The candidate subject must precede the verb, while the candidate object must follow the predicate pattern. For the sentence in Figure 1 the candidate subject is "*Nakamura*", while the candidate object is "*il quartier generale di il Kyokushin Karate*"[6].

After identifying the candidate subject and object, the triple is accepted only if both the subject and the object have a syntactic relation with the verb. In particular, one of the tokens belonging to the subject/object must have a dependent relation with a token of the verb pattern.

More details about the unsupervised extraction of triples are reported in Cassotti et al. (2021).

3.1 Self-Training

Using the unsupervised approach, we obtain 3,562,803. We randomly select a subset of 200 triples for which the predicate occurs at least 20 times. Then, each triple is annotated by two experts as relevant (valid) or not-relevant. Details on this dataset and the results of the annotation process are reported in (Cassotti et al., 2021). For the self-training, we select only triples in which the two experts agree. Finally, we have a set of 137 triples that we call L.

From the whole set of 3.5M triples, we randomly select the 1% of unlabeled triples in which the predicate occurs at least 20 times. This subset is denoted as U. The set L is split in two subsets: L_t for training and L_v for validation. In particular, L_t is used as the initial dataset for the self-training procedure, while L_v is used for setting the initial parameters' values of the learning algorithm.

As a preliminary step, we search for the best parameters using L_t for training and L_v for validating the performance. We use the macro-averaged F1 score since our dataset is highly unbalanced: the 82% of the triples are labelled as relevant.

The self-training process works as follow:

1. from the set U, we randomly select p triples;
2. we train a supervised model using labeled triples in L_t;
3. the p triples are labeled using the trained model, and a confidence score is assigned to each classified triple;
4. the triples with a confidence score higher or equal to a threshold t are added to L_t by maintaining classes balance in L_t. If the classifier does not provide a confidence score, all instances labeled as valid are included in L_t;
5. if U contains at least p triples go to step 1 otherwise ends. The self-training loop can also be terminated if a specific number of iterations is reached.

The resulting set of labeled triples L_t is used to train the final model, which is employed to classify all the triples extracted using the unsupervised approach.

More details about both the parameters' values and the training algorithm are reported in Section 4.

3.2 Supervised Approach

For both the self-training and the classification of triples, we exploit algorithms provided by LibLinear[7]. In particular, we use both logistic regression and support vector classification: the former can provide a confidence score, while the latter cannot.

The set of features is selected by taking into account the supervised approaches already developed for English. In particular, we use:

- the PoS-tags occurring into the subject, object, and predicate;
- the sequence of PoS-tags that compose the predicate. This feature is also computed for both the subject and the object;
- the n-gram that composes the predicate;
- the set of dependencies that link the subject to the predicate;
- the set of dependencies that link the object to the predicate.

The C value of the learning algorithm is determined by performing a grid search using L_t for training and L_v for validating. Due to the small size of the original set L, we perform a 50/50 split. More details are reported in Section 4.

[6] It is important to note that UDPipe splits the articulated preposition "del" in "di:ADP" and "il:DET".

[7] https://www.csie.ntu.edu.tw/ cjlin/liblinear/

Method	C	P_0	R_0	$F1_0$	P_1	R_1	$F1_1$	F1
S_{log}	10	.54	.58	.56	.91	.89	.90	.62
S_{svc}	8	.60	.75	.66	.94	.89	.92	.73

Table 2: Results of the grid search.

Method	Size	P_0	R_0	$F1_0$	P_1	R_1	$F1_1$	F1	$\Delta\%$
S_{log}	15,771	.88	.58	.70	.92	.98	.95	.74	19.35%
S_{svc}	19,545	.60	.50	.55	.90	.93	.91	.60	-17.81%

Table 3: Results of the self-training approach.

4 Evaluation

The goal of the evaluation is twofold: 1) measure the performance and the contribution of the self-training; 2) evaluate the quality of the extracted triples. For the first goal, we evaluate how the new instances added to the initial set of training affect the performance. For the second goal, we manually annotated a small subset of extracted triples in order to evaluate their quality.

4.1 Evaluate Self-Training

The first step is to determine the best parameters for the learning algorithm. We use two algorithms: L2-regularized logistic regression (S_{log}) and L2-regularized L2-loss support vector classification (S_{svc}). For both algorithms, we perform a grid search for selecting the best value for the parameter C. Results of the grid search is reported in Table 2. In the table, we report the best C value for each approach. We denote with 0 the class of not-relevant triples, while 1 denotes relevant ones. F1 refers to macro-average F1. Results show that classifiers have poor performance in recognizing the class 0 since the dateset is both small and unbalanced.

We perform two self-training steps (one for each learning algorithm) using $p = 1,000$ and 20 as the number of maximum iterations. For the logistic regression, we set 0.85 as threshold. After the self-training step, we obtain a new training set which contains new instances. Table 3 reports for each learning approach the size of the new training set and the performance computed on the validation set. Moreover, the last column reports the increment of F1 with respect to the performance obtained before the self-training.

Experiments using self-training show that S_{log} is able to improve (+19%) its performance, while self-training has a negative impact on S_{svc} performance (-18%). Probably, this is due to the fact that it is not possible to set a threshold for selecting good classified instances during the self-training when the S_{svc} is involved. After observing the overall performances in both Tables 2 and 3, we select as training set for extracting triples the one obtained by S_{log} during the self-training. S_{log} is also able to both overcome the performance of S_{svc} obtained without self-training and achieve also an improvement in $F1_0$.

After the extraction and classification process, we obtain 2,974,374 triples[8] as reported in Table 4. The original set of triples extracted from the unsupervised approach was 3,562,803, this means that the 16.52% of unsupervised triples was classified as not-valid. Table 4 reports also information about the number of distinct subjects, objects, and predicates for both the unsupervised and supervised datasets. The supervised dataset is released in the same JSON format described in Cassotti et al. (2021).

4.2 Evaluate Triples

For the evaluation, we follow the same methodology proposed in Cassotti et al. (2021). In particular, we sample a subset of 200 triples from the final set of classified triples. The triples selected are the ones for which the predicate occurs at least 20 times. Then, each triple is annotated by two experts as relevant (valid) or not-relevant. We used Cohen's Kappa coefficient (K) to measure the pairwise agreement between the two experts. K is a more robust measure than simple percent agreement calculation since it takes into account the agreement occurring by chance. Higher values of K correspond to higher inter-rater reliability. Open

[8]The triples are available on Zenodo: `https://zenodo.org/record/5655028`. The triples obtained by the unsupervised approach are available here: `https://doi.org/10.5281/zenodo.5498034`.

Dataset	#triples	#dist. subj	#dist pred	#dist obj
unsupervised	3,562,803	1,298,481	269,551	2,030,742
supervised (S_{log})	2,974,374	1,189,648	241,053	1,720,348

Table 4: Dataset statistics.

Dataset	#valid (exp 1)	#ratio (exp 1)	#valid (exp 2)	#ratio (exp 2)	Kappa C.
unsupervised	115	0.64	161	0.81	0.24
supervised (S_{log})	158	0.79	163	0.82	0.63

Table 5: Results of the annotation process.

IE task lacks a formal definition of triple relevance thus for the annotation process, we adopt the concept of triple relevance reported in (Stanovsky and Dagan, 2016) that is based on assertiveness, minimalism, and completeness. This ensures that: the triples extracted still enclose the semantics of the original sentence (assertiveness), each element of the triple is as compact as possible without any unnecessary In our evaluation, we decide to give less weight to minimalism and focus more on the extraction completeness. After the annotation, we compute the ratio of relevant triples (column #ratio in Table 5) for each dataset and expert. Specifically, the ratio is computed dividing the number of triples annotated as relevant by the number of sampled triples.

Results of the evaluation are reported in Table 5, where also the previous results on the set of unsupervised triples is reported. It is important to highlight that the two datasets are not directly comparable since they are composed of different triples. In particular, a small subset of the unsupervised dataset is used to train the supervised one as explained in Section 3. Cohen's kappa coefficient for each dataset is provided in the last column of Table 5.

We obtain a good result in terms of number of valid triples. In particular, the supervised model provides a set of triples that improve the agreement between annotators. The supervised approach removes noisy and ambiguous triples since the initial subset L_t used for self-training contains only triples for which the annotators agree.

In this task, it is not always possible to compute standard metrics such as recall since it is not easy to determine the total number of valid triples due to the task's "open" nature. As future work, we plan to extend the number of manually annotated triples for performing a more rigorous evaluation and comparison of different information extraction methods for Italian.

5 Conclusions and Future Work

We propose a self-training strategy for implementing a supervised open information extraction system for the Italian version of Wikipedia. Our approach exploits a small set of manually labeled triples for expanding the training set. We integrate this system into WikiOIE, which is a framework for open information extraction on Wikipedia dumps. WikiOIE exploits UDPipe as a tool for processing and annotating the text and can be extended by adding several information extraction approaches.

We perform an extensive evaluation for measuring the impact of self-training on the overall classification performance. Results prove that self-training is able to improve the classification performance and help to identify not-relevant triples.

Finally, we sampled a subset of extracted triples, evaluated by two experts. The number of relevant triples increases when the self-training strategy is used by also improving the agreement between annotators.

As future work, we plan to extend the evaluation to a larger scale study, exploit several learning algorithms, and explore the application of the approach to other languages.

Acknowledgments

This research was partially funded by the INTERREG-MEDITERRANEAN Social and Creative project, priority axis 1: Promoting Mediterranean innovation capacities to develop smart and sustainable growth, Programme specific objective 1.1 To increase transnational activity of innovative clusters and networks of key sectors of the MED area (2019-2022).

References

Giuseppe Attardi. 2015. Wikiextractor. https://github.com/attardi/wikiextractor.

Pierluigi Cassotti, Lucia Siciliani, Pierpaolo Basile, Marco de Gemmis, and Pasquale Lops. 2021. Extracting Relations from Italian Wikipedia using Unsupervised Information Extraction. In Vito Walter Anelli, Tommaso Di Noia, Nicola Ferro, and Fedelucio Narducci, editors, *Proceedings of the 11th Italian Information Retrieval Workshop 2021 (IIR 2021)*. CEUR-WS. http://ceur-ws.org/Vol-2947/paper2.pdf.

Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. 2008. Open information extraction from the web. *Commun. ACM*, 51(12):68–74.

Raffaele Guarasci, Emanuele Damiano, Aniello Minutolo, Massimo Esposito, and Giuseppe De Pietro. 2020. Lexicon-Grammar based open information extraction from natural language sentences in Italian. *Expert Syst. Appl.*, 143.

Patrick Hohenecker, Frank Mtumbuka, Vid Kocijan, and Thomas Lukasiewicz. 2020. Systematic comparison of neural architectures and training approaches for open information extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8554–8565, Online, November. Association for Computational Linguistics.

Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. 2012. Open Language Learning for Information Extraction. In Jun'ichi Tsujii, James Henderson, and Marius Pasca, editors, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL*, pages 523–534, Jeju Island, Korea, 7. ACL.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado, June. Association for Computational Linguistics.

Gabriel Stanovsky and Ido Dagan. 2016. Creating a Large Benchmark for Open Information Extraction. In Jian Su, Xavier Carreras, and Kevin Duh, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016*, pages 2300–2305, Austin, Texas, USA, 11. The Association for Computational Linguistics.

Milan Straka and Jana Straková. 2017. Tokenizing, POS Tagging, Lemmatizing and Parsing UD 2.0 with UDPipe. In Jan Hajic and Dan Zeman, editors, *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada, 8. Association for Computational Linguistics.

Fei Wu and Daniel S. Weld. 2010. Open Information Extraction Using Wikipedia. In Jan Hajic, Sandra Carberry, and Stephen Clark, editors, *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 118–127, Uppsala, Sweden, 7. The Association for Computer Linguistics.

Mohamed Yahya, Steven Whang, Rahul Gupta, and Alon Y. Halevy. 2014. ReNoun: Fact Extraction for Nominal Attributes. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 325–335, Doha, Qatar, 10. ACL.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts, USA, June. Association for Computational Linguistics.