Minimally Supervised Event Argument Extraction using Universal Schema

Benjamin Roth **Emma Strubell** **Katherine Silverstein** **Andrew McCallum**
School of Computer Science
University of Massachusetts, Amherst
beroth,strubell,ksilvers,mccallum@cs.umass.edu

1 Introduction

The prediction of events and their participants is an important component of building a knowledge base automatically from text. Typically, the events of interest are domain-specific and not known in advance, and so it is often the case that little or no training data is available to learn the appropriate predictors. In this work, we propose a technique for distantly supervised event argument extraction based on matrix factorization using *Universal Schema* [1].

The Universal Schema approach to event argument extraction uses no previously annotated training data. Instead, the starting point is solely the event argument slot definitions. We write surface patterns to capture the intuitive understanding of each event role, limiting the time spent writing patterns to five minutes per role. Those patterns are then expanded using matrix factorization, in order to increase recall: A matrix is built with contextual patterns from the whole source corpus, so that similarity to the seed patterns can be leveraged to obtain new patterns for event role prediction. In this way, we leverage a large text corpus to build a matrix that captures meaningful correlations between per-entity surface features and signals from manual seed patterns, requiring little human intervention.

On the TAC 2014 Event Argument Extraction pilot data, our method improves both recall and F1-score over a baseline using manual patterns only, resulting in better coverage of event arguments.

2 Task Description

We apply our method to the TAC 2014 Event Argument Extraction (EAE) task[1]. In this task, we are given a fixed set of event types such as MOVEMENT.TRANSPORT or JUSTICE.SENTENCE which each have a set of typed entity arguments. For example, the MOVEMENT.TRANSPORT event has six possible arguments, including ARTIFACT, the person, weapon or vehicle being transported, and AGENT, the person, organization or geopolitical entity that is transporting the ARTIFACT. We view this task as a slot-filling problem, where each event-argument pair corresponds to a single slot to be filled by an entity of the appropriate type. The TAC EAE task description thus describes just under 100 possible event slots. Given a diverse corpus of text drawn from newswire, discussion forums, and web documents, our goal is to correctly fill as many of these

[1]Official task description available at: http://www.nist.gov/tac/2014/KBP/Event/guidelines.html

	ARG transported	ARG shipped	ARG moved	TRANSPORT-AGENT
france:DOC032014	1		1	1
usa:DOC012000	1	1		1
australia:DOC121990		1	1	?

Figure 1: Simplified example of a training matrix for three documents and an event role TRANSPORT-AGENT with the manual seed pattern `ARG transported`. `ARG shipped` and `ARG moved` are potential candidates for bootstrapped patterns.

event argument slots as possible. We can evaluate our performance in terms of F1-score using the annotated event argument extraction data collected via the TAC 2014 EAE Pilot; at the same time, since there does not exist dedicated training data, we base our training on seed patterns.

3 Method

Our approach to minimally supervised event argument extraction is to use a small set of manual seed patterns to learn a much larger set of surface patterns that signify event argument slot fillers. We learn these correlations between patterns and slots by using embeddings of entities, patterns and event arguments to score context patterns, which we then use to extract event argument slot-fillers from text. This approach is inspired by the *Universal Schema* approach to matrix factorization for binary relation extraction [1].

3.1 Universal Schema Matrix Factorization

Universal Schema matrix factorization works by embedding each row r (entity) and column c (relation or pattern) of the matrix into a k-dimensional latent representation $\mathbf{v}_r$ and $\mathbf{w}_c$, respectively, where k is a fixed input parameter. We use matrix factorization based on logistic regression [2], where each binary cell is obtained by applying the logistic function $\sigma(\cdot)$ to the scores resulting from the factorization into $\mathbf{v}_r$ and $\mathbf{w}_c$.

$$\theta_{r,c} = \sum_i v_{r,i} w_{c,i}$$
$$x_{r,c} = \sigma(\theta_{r,c})$$

In other words, each cell $x_{r,c}$ is modeled as a Bernoulli random variable with natural parameter θ equal to the product of low-rank vectors $\mathbf{v}_r$ and $\mathbf{w}_c$. We learn these embeddings using ℓ_2-regularized stochastic gradient descent with Bayesian personalized ranking (BPR) updates [3].

3.2 Training Data Generation

We begin by generating a small set of manual seed patterns. An example seed pattern for the slot MOVEMENT.TRANSPORT-AGENT is `ARG transported`, where `ARG` is a stand in for the tagged named entity. For each slot, we limit our seed patterns to those that can be written by an individual within a given time limit. For example, generating patterns for the hundred slots defined in the TAC EAE task with a time limit of five minutes requires a maximum of 8 total hours of human supervision.

In our approach to constructing the training matrix we start from the premise that each entity takes on exactly one event role per document, but is not necessarily associated with that role in other documents. We encode this by grouping context information according to *document-specific entities*, i.e. entity names marked by a document id, which form the rows of the universal schema training matrix. We only consider document-specific entities that occur at least twice, so that information can be transferred across contexts. By using

score	event role	pattern
0.9995	`Justice.Extradite Destination`	extradited to ARG
0.9972	`Justice.Sentence Defendant`	ARG was sentenced
0.9970	`Justice.Convict Defendant`	ARG ? convicted in
0.9953	`Justice.Sentence Defendant`	ARG ? sentenced in
0.9949	`Justice.Extradite Person`	to extradite ARG
0.9940	`Movement.Transport Destination`	returned to ARG
0.9934	`Justice.Charge-Indict Defendant`	ARG pleaded not
0.9931	`Justice.Charge-Indict Defendant`	Police charged ARG
0.9925	`Movement.Transport Destination`	had traveled ? ARG

Figure 2: Examples of patterns found and scored by universal schema.

document-specific entities, information is shared across documents via the pattern representations (column vectors), while the rows do not aggreagate across documents.

The matrix has two types of columns: First, contextual pattern columns, which are simply bigrams in a sliding window of size 4 around the document-specific entities, with the entity position marked by a wildcard (`ARG`). Tokens within the sliding window that fall between the bigram and entity are skipped over (indicated by `?`). Second, one column for each event role. The matrix cells for the contextual pattern columns are filled if an entity co-occurs with a particular pattern, and the cells for the event role columns are filled if a seed pattern matches an occurrence of the entity in the respective document. See Figure 1 for a simplified example of the training matrix.

3.3 Pattern Bootstrapping

The training matrix is factorized, and vector embeddings are obtained for the document-specific entities, the context patterns, and the event roles. For each context pattern, the similarity to the event roles is measured by taking the cosine distance between their embeddings. The top-n context patterns with the highest similarity are used to predict relations on the test data, see Figure 2 for examples of high-ranked patterns.

For prediction on the test data, patterns exceeding the similarity threshold are compared against contexts around event argument candidates (i.e. named enitities of the correct type). This way, each prediction is associated with a pattern match position in a document, and potentially several event roles can be predicted for the same entity.[2]

4 Experimental Results

We evaluate our system using a subset of the annotated data from the TAC 2014 EAE Pilot[3]. Specifically, we limit our evaluation to those entity mentions that the NLP pipeline (tokenization, sentence segmentation and named-entity tagging) recognized in the given document; the preprocessing steps were performed using the FACTORIE [4] NLP tools.

[2]The local predictions of the bootstrapped patterns (or the pattern vectors directly) could be used in a less local predictor that jointly optimizes over several event arguments in a document. However, since our seed expansion method is motivated by settings where training coverage is limited, it would be challenging to train such a joint predictor on incomplete training data.

[3]Note that this task was newly introduced in 2014, and no official results on this data are available yet for comparison.

Method	Precision	Recall	F1-score
Seed	**43.84**	6.93	11.96
USchema (1k)	31.94	4.98	8.61
USchema (10k)	16.51	8.01	10.79
Seed + USchema (1k)	39.13	9.74	**15.60**
Seed + USchema (10k)	19.84	**11.04**	14.19

Table 1: Precision, recall and F1 score of our system evaluated on the TAC 2014 Event Argument Extraction Pilot data.

The task in TAC EAE is to predict event *arguments* only, i.e. it is not required to connect the predicted arguments and specify which of them would together form an event. Thus we compute precision as the number of correctly labeled mentions out of all the mentions that were labeled with an event role, recall as the number of correctly labeled mentions out of all the entity mentions that we found, and F1 as the harmonic mean of precision and recall. Overall, there are 60 documents in the TAC EAE Pilot test collection, with 685 entities that were detected by our named entity recognizer and assigned an event role by the annotators.

Our experiments consist of the following:

- **Seed** This setup uses only manual seed patterns.
- **USchema1k (USchema10k):** Universal schema patterns only, using top 1 000 (top 10 000, respectively) universal schema patterns.
- **Seed+USchema1k (Seed+USchema10k):** This setup is a merger of matching the seed patterns and the top 1 000 (top 10 000, respectively) induced patterns from universal schema.

Our results are listed in Table 1. For the seed patterns, the large difference between recall and precision is striking. With universal schema the ratio between recall and precision is better controllable; At the same time, the overall lower precision makes it hard for Universal Schema to beat the seed pattern F1-score on its own. In combination with the seed patterns, one can see that the Universal Schema patterns contribute complementary information, leading to the overall best recall and F1-scores in our experiments.

5 Discussion

While the overall scores of our initial experiments remain in a low range, we think that pattern expansion using manual seed patterns and matrix factorization is especially interesting for use cases where increasing recall is critical, and where only limited human annotation resources are available.

Our experiments show that while recall is the bottleneck for seed patterns, maintaining reasonably high precision is currently the main difficulty for the induced Universal Schema patterns. There are several potential sources of noise that would be interesting to investigate:

- The assumption of one role per entity and document: This is similar to the distant supervision assumption in relation extraction [5]. However, while for relations the default meaning of two co-occurring entities is often biased towards a particular fact (e.g. *Honulu* is mentioned together with *Barack Obama* mostly as his birth place), when entities (e.g. *Honulu*) are looked at in isolation, the range of possible events is much wider.
- Noisy contextual patterns: When generating bigrams in a sliding window, for each entity both potentially meaningful patterns (e.g. those patterns containing content words), as well as noisy

patterns (e.g. containing function words or other entities) are collected. It seems promising to explore better contextual representations, possibly based on dependency patterns and focused around content words.

6 Conclusion

The Universal Schema approach to event argument extraction leverages few seed patterns and a large corpus to score contextual patterns with respect to their similarity to event roles. The approach increases recall without much loss in precision compared to using the manual seed patterns alone, leading to an improved F1-score on the TAC Event Argument Extraction task in a minimally supervised setting.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval, in part by DARPA under agreement number FA8750-13-2-0020, in part by IARPA via DoI/NBC contract #D11PC20152, and in part by NSF grant #CNS-0958392. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

References

[1] Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. Relation extraction with matrix factorization and universal schemas. In *Joint Human Language Technology Conference/Annual Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL '13)*, 2013.

[2] Michael Collins, Sanjoy Dasgupta, and Robert E. Schapire. A generalization of principal component analysis to the exponential family. In *Neural Information Processing Systems (NIPS)*, 2001.

[3] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 452–461. AUAI Press, 2009.

[4] Andrew McCallum, Karl Schultz, and Sameer Singh. FACTORIE: Probabilistic programming via imperatively defined factor graphs. In *Neural Information Processing Systems (NIPS)*, 2009.

[5] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics, 2009.