Universal Schema for Slot Filling and Cold Start: UMass IESL at TACKBP 2013

Sameer Singh, Limin Yao, David Belanger, Ari Kobren, Sam Anzaroot, Mike Wick, Alexandre Passos, Harshal Pandya, Jinho Choi, Brian Martin, Andrew McCallum
School of Computer Science
University of Massachusetts, Amherst
mccallum@cs.umass.edu

Abstract

We employ universal schema for slot filling and cold start. In universal schema, we allow each surface pattern from raw text, and each type defined in ontology, i.e. TACKBP slots to represent relations. And we use matrix factorization to discover implications among surface patterns and target slots. First, we identify mentions of entities from the whole text corpus and extract relations between entity pairs to construct a knowledge base. Finally, we query this knowledge base to produce our submissions for slot filling and cold start.

1 Introduction

Due to its importance in information extraction pipelines, relation extraction has attracted attention in TAC KBP for a number of years as the Slot Filling track, and as a vital sub-task of the recently-introduced Cold-Start track. Many existing relation extraction techniques (Liu and Zhao, 2012; Min et al., 2012; Roth et al., 2012) of the following: (1) matching of textual mentions to the query mentions, often utilizing information retrieval techniques, (2) the extraction of the context around the mention, (3) identification of the relation being expressed in the context (if any), (4) aggregation of the individual classifications to resolve redundancies and inconsistencies. Although this overall architecture is common, the approaches differ considerably in the approach used for matching, extraction, identification (using sophisticated rules, lexicons, classifiers, graphical models, etc.) and aggregation.

There are a number of disadvantages, unfortunately, inherent in such systems. The foremost problem is obtaining training data for learning the extractors, as the amount of training data available as part of TACKBP is quite inadequate. Many systems use manually-created seed patterns to bootstrap a labeling (often in combination with the TACKBP training data), followed by induction to expand the set of patterns for the extractors. More recently, *distant supervision* approaches that use the relations in external knowledge bases such as Freebase or Wikipedia (manually aligned to the TACKBP schema) have achieved huge improvements (Roth et al., 2012). However, the resulting training data from both seeded patterns and distant supervision is often noisy due to the assumptions made when expanding the labeled data. Further, it is often not the case that a single pattern in a sentence is indicative of a relation between two entities, and instead, multiple relations/patterns may in combination imply a relation that is not evident when the patterns are observed separately. As a simple example, consider that recognizing *Apple Inc.* is headquartered in *California* might require an extraction that provides evidence that *Apple* is located in *Palo Alto*, and a separate extraction that indicates *Palo Alto* is in *California*. Along with requiring inference of relations across multiple extractions, this example further requires extraction of slots for non-query entities (*Palo Alto* in this example) and also would benefit from leveraging the implicit implications amongst relations.

In this work, we introduce a novel approach to the TAC KBP slot filling and cold start tasks that produces answers jointly on all entities and relations, rather than on a query-by-query or a per-extraction basis. From a large collection of documents consisting of both training and test KBP documents, we construct a knowledge-base over all the entities (including many that do not appear in the queries or the reference KB). The relations are extracted by using the matrix factorization-based universal schema approach (Riedel et al., 2013) that learns the correspondences between co-occurring entity pairs, observed textual patterns, and the labeled relations, as part of a joint optimization over training and test data in order to predict the unknown relations. Due to the nature of the factorization, the model is able to not only learn the implications between text patterns and relations, but is also able to leverage soft associations between patterns, between relations, and between entities, to generalize the observed text patterns to new entities and extracted relations in the knowledge base. In order to provide our slot filling submission, we query the extracted knowledge base with the query entities (the complete knowledge base is our cold start submission).

The overall system consists of the following architecture. First, we process the text corpus using a natural language processing pipeline. This includes part-of-speech tagging, dependency parsing, mention finding and coreference, as described in section 2. Second, we extract binary relations between pairs of entities using universal schema (section 3). In order to predict the string-valued slots (such as `alternate_name`) that are not supported by *universal schema*, we use manually constructed rule-based heuristics (Section 5). The extractions from two steps are combined to produce an internal knowledge base about the corpus.

Since this is our first year at TAC KBP Slot-filling and Cold Start, we do not expect to outperform the existing approaches. In Section 4, we provide an investigation of our 2013 predictions, and demonstrate the various associations that our model learns. Using a provenance-agnostic metric, we obtain 12.5% F1 for the entity-valued slots and 19.3% for the string-valued slots. Further, we observe that our universal schema extraction method achieves high recall, but suffers from low precision. Conversely, our rule-based extractors produce high precision and low recall. Using the official metrics (that take the provenance into account) we achieved 13.7% F1 for slot filling.

2 Data Processing Pipeline

Figure 1 outlines our processing pipeline, which is almost identical for both our slot filling and cold start systems (differs only in the set of documents and queries that are fed into the pipeline, and the filtering, if any, of the resulting knowledge base).

2.1 Corpus Selection

Recall that universal schema performs relation extraction jointly on train and test data. Therefore, we need to assemble a sub-collection of the TAC KBP training documents that are relevant either for training or for the submission. For slot filling, we select 120K documents by performing query expansion on the 2013 queries (both training and test) using the Galago search engine on all the TAC KBP 2013 documents (Cartright et al., 2012), used for entity linking in TAC KBP 2012 (Dietz and Dalton, 2012). For the cold start track, we extend this collection with the cold start source documents.

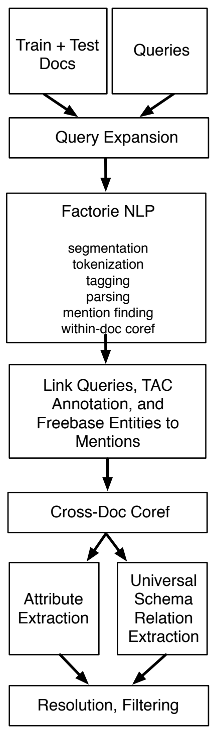


Figure 1: Description of the overall pipeline.

2.2 Natural Language Processing

We employ the open-source FACTORIE package (McCallum et al., 2009) as our natural language processing pipeline for processing the documents. This software package includes a number of state-of-the-art low-level NLP processing tools, including part-of-speech, dependency parsing, and named entity recognition (NER), all based on machine learning models.

2.3 Mention Finding and Entity Discovery

Mention finding and entity discovery play a critical role in our pipeline, since without accurate entities, we cannot build a high quality knowledge base to answer the TACKBP slot filling and cold start queries. Our set of mentions consist of the detected named entities and pronouns, as well as all queries and annotations. We use the FACTORIE within-doc coreference system to cluster mentions for each document into anaphoric sets. This system performs greedy left-to-right grouping of mentions with features based on Bengtson and Roth (2008).

To assemble the cross-document entities for the knowledge base, all the within-document entities, along with entities from the reference KB, need to be aggregated. The observed *string names* of the within-document and the reference entities are normalized by removing prefixes and suffixes such as titles and honorifics, followed by a string-matching based grouping. For cold-start, we also include relation annotations from Freebase (described in the next section) for which we perform trivial linking of Freebase entities and our cross-document entities based on available names and redirects in Freebase.

3 Slot Filling Using Universal Schema

We cast slot filling as relation extraction, considering each slot as a relation instance of the query and the slot value. We employ universal schema for slot filling (Riedel et al., 2013). Instead of classifying entity pairs into pre-defined relation types, universal schema takes both the surface patterns and pre-defined types as relations, and uses matrix factorization to discover implications among them. This approach can leverage unlabeled data, while avoiding brittle alignment errors in the distant supervision methods that have been popular in past TAC/KBP competitions i.e. entity pairs labeled in the text or from existing knowledge bases.

We fill a matrix with relation instances, where each row corresponds to an entity pair and each column to a relation, including surface patterns and slot names. The surface patterns include the sequence of words between two entity mentions, the dependency path connecting two mentions and so on. We only include TACKBP slots that describe relations between two entities (*entity-valued* slots).

Our goal is to predict target slots for the entity pairs. The task is analogous to collaborative filtering. In collaborative filtering, items are recommended to users based on collecting many users' ratings about the items. For example, if both users X and Y like item A, and user X also likes item B, it is likely that user Y likes item B as well. In our scenario, an entity pair corresponds to a user; a relation (pattern or slot name) corresponds to an item; and an observed cell (i.e. entity pair is either observed with the surface pattern, or is annotated with having the slot name) corresponds to a positive rating by the user for the item. By collecting information about observations of other entity pairs, we can "recommend" relations that hold for the entity pair of interest.

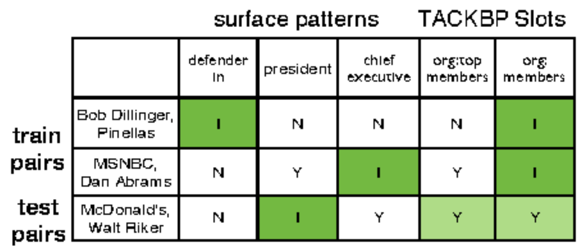
In our matrix factorization approach, we embed each entity pair and relation as latent vectors $\mathbf{a}_e$ and $\mathbf{v}_r$ in a K-dimensional space, respectively. Each dimension is a component (c). Since squared loss is not appropriate for discrete data, a logistic regression version of matrix factorization is a better choice for our binary data (Collins et al., 2001). Thus we have:

$$\begin{aligned}\theta_{e,r} &= \sum_c a_{e,c} v_{r,c} \\ x_{e,r} &= \sigma(\sum_c a_{e,c} v_{r,c}) \\ \sigma(\theta) &= \frac{1}{1+\exp(-\theta)}\end{aligned}$$

The first formula is factorizing a matrix into multiplication of two matrices. We apply a logistic function to this θ score to model a binary cell. This has a probabilistic interpretation: each cell is drawn from a Bernoulli distribution with natural parameter θ. Note that an observed pattern or slot r between an entity pair e is encoded by $\sigma(\theta_{e,r}) \equiv 1$.

Figure 2 shows an illustration of universal schema. In training, we learn low dimensional embeddings for entity pairs and relations. We can complete the whole matrix using these embeddings. For slot filling we only need to query specific rows that have one argument from the TACKBP queries. We predict top ranked slots based on the scores. For example, consider the query entity "McDonald's"; we have one pair (McDonald's, Walt Riker), and predict *org:top_members* with confidence 0.81 and *org:members* with confidence 0.60. In post processing, we use the prediction probabilities and basic consistency rules to filter these predictions. We describe how we leverage geometric interpretation of our embeddings to extract the provenance at the end of section 4.2.

In the following, we describe how to train our model. We learn low dimensional representations for entity pairs and relations by maximizing the



		surface patterns			TACKBP Slots	
		defender in	president	chief executive	org:top members	org: members
train pairs	Bob Dillinger, Pinellas	1	N	N	N	1
	MSNBC, Dan Abrams	N	Y	1	Y	1
test pairs	McDonald's, Walt Riker	N	1	Y	Y	Y

Figure 2: **Illustration of universal schema.** Each row represents an entity pair and each column a relation. A cell is true if we observe that an entity pair co-occurs with a relation in text or in TACKBP annotation (green cells marked "1"). Our goal is to predict whether a relation (slot) is true for particular entity pairs. The last row is an example test pair, we predict two slots for this pair (light green cells marked "Y").

log likelihood of the observed cells under the probabilistic model above. Notice that in our training data we only observe positive cells and have no accurate data on which relations do *not* hold for an entity. However, learning requires negative training data. We address this issue by sampling unobserved relations for an entity based on their frequencies in the whole dataset and treating them as negative. The joint probability of all cells is defined as:

$$\prod_{cell} p(x_{cell} = 1)^{\delta(x=1)} (1 - p(x = 0))^{\delta(x=0)}$$
$$= \prod_{cell} \sigma(\theta)^{\delta(x=1)} (1 - \sigma(\theta))^{\delta(x=0)}$$

For simplicity, we elide the subscript of each cell, and $\delta(x = 1)$ stands for the number of positive cells.

To simplify the joint probability, we can represent negative cells as positive cells by choosing a different natural parameter. Thus the joint probability becomes $\prod_{cell} \sigma(\theta)$. For simplicity, we use Θ to represent all the parameters. Adding a prior for the parameters, we can write the log likelihood as

$$\log \sum_{cell} \sigma(\theta) - \Lambda_\Theta ||\Theta||^2.$$

The gradient with respect to θ is:

$$(1 - \sigma(\theta))\nabla\theta - \lambda\theta$$

Taking gradients of θ with respect to the parameters, we obtain:

$$\begin{aligned} \nabla a_{e,c} &= v_{r,c} \\ \nabla v_{r,c} &= a_{e,c} \end{aligned}$$

We use stochastic gradient optimization to effectively deal with the large scale of our matrices. In each iteration, we traverse random permutations of all training cells, randomly sample 3 to 5 negative cells for each training cell, and update the corresponding $\mathbf{a}_e$ and $\mathbf{v}_r$ vectors for the positive and negative cells based on their corresponding gradients.

We update the parameters of a positive cell (e, r) using the following formulas, iterating over each component c, with learning rate l:

$$\begin{aligned} a_{e,c} &+= l(1 - \sigma(\theta))v_{r,c} - \lambda a_{e,c} \\ v_{r,c} &+= l(1 - \sigma(\theta))a_{e,c} - \lambda v_{r,c} \end{aligned}$$

Likewise for a negative cell, we update the parameters using:

$$\begin{aligned} a_{e,c} &+= l(0 - \sigma(\theta))v_{r,c} - \lambda a_{e,c} \\ v_{r,c} &+= l(0 - \sigma(\theta))a_{e,c} - \lambda v_{r,c} \end{aligned}$$

When querying a specific cell, we calculate its score as $x_{e,r} = \sigma(\theta_{e,r})$.

4 Experiments

In this section, we discuss our experiments on slots extracted by universal schema.

4.1 Observed Cells

Our approach learns implicature among the columns of the matrix, which consist of observed surface patterns and relations.

We use the following *features* as the columns for surface patterns. We use the dependency path of the sentence and the word sequence between two entity mentions as surface patterns. Additionally, we use conjunctions of trigger words and entity types (see Table 1 for examples). A dependency path is a concatenation of dependency relations and words on the path connecting two entity nodes, as shown in the table. The suffix ":BACK" signals that the second argument of the entity pair comes before the first in the sentence. The trigger is usually taken as the root of the path connecting the two nodes.

For the relations, the columns consist of slot names as defined by a schema. The TAC KBP

Feature Type	Example
Dep-path	↑pobj↑in↑prep↑die↓nsubj↓
Words	die in:BACK
Type-Words-Type	ORG-die in:BACK-PER
Type-Trigger-Type	ORG—die—PER

Table 1: Surface patterns used in universal schema. These patterns are extracted for entity pair (Abbey Church, Father Daniel) from sentence "Father Daniel died in the Abbey Church at Saint Anselm"

slot names, such as per:spouse or org:founded_by, appear as columns with observed cells for entity pairs that are annotated either in TAC KBP training documents or in the reference KB. These columns are crucial for learning from surface patterns, and for predicting a TAC KBP relation between any entity pair. For cold start, we also include a number of entity pairs from Freebase, including additional columns that denote relations as per Freebase schema (allowing our system to learn the associations between surface patterns, TAC relations, and the Freebase relations).

4.2 Embeddings

To interpret the embeddings of the surface patterns, we calculate the cosine similarity between vectors of a TAC relation column and every column. We list the top ranked patterns with respect to each target slot in Table 2. For simplicity, we translate the dependency path to word sequence. To generalize the patterns, we replace tokens of part-of-speech tag "NNP" with their tags. For example, in pattern "replace NNP NNP as face of," those two "NNP" tokens may stand for a person name. The suffix ":INV" indicates that the two arguments are in inverse position. Some patterns are augmented with entity types of the two arguments. Slots that are similar to the target slots are also shown, for example, *per:organizations_founded* is indicative of slot *employee_of*.

We observe here that our approach can learn diverse and accurate patterns that are indicative of the target slots. For example, we extract patterns that contain "ex-wife," "hubby," "file divorce" for *spouse*, patterns that include "pay by," "resign from" for *employee_of*. We also analyze errors made by our approach. Some errors are caused by incorrect dependency path extracted from the text. For example, pattern "X into a career to Y" is closely related to *spouse* due to incorrectly extracting the path for entity pair (Lucinda, Robert Morgenthau) from the sentence "Lucinda into a Pulitzer Prize-winning career in journalism and marriage to New York District Attorney Robert Morgenthau." Since annotation data is limited, this pattern is in the top ranked list for predicting *spouse*. Some errors are that our model cannot distinguish two slots from each other. For example, some surface patterns that are similar to *employee_of* are also in the top ranked list for *schools_attended*. Generally speaking, *schools_attended* should be a sub relation of *employee_of*. Our model does learn patterns specific to schools, such as "be junior at," "his/her college in," and "graduate of."

Embedding all relations, both from the target schema and from surface patterns, in a common euclidean space allows us to provide provenance for our predictions. First, we keep the pairwise similarity between the embeddings for every pair of columns cached. Suppose we predict that X is related to Y by the TAC KBP relation *employee_of*. Then, we consider all surface pattern columns that were observed for the pair (X,Y), and find the one that has an embedding most similar to the embedding for *employee_of*. We use the sentence that produced the observed cell for this column as our provenance.

4.3 Results

We evaluate our predictions on 2013 data without considering provenance. A prediction is correct if it matches the annotation. Table 3 lists precision, recall and F1 measures for different slots. We can see that our approach achieves higher recall.

5 Attribute Relation Extraction

Some slots fillers are string-valued (i.e. attribute-value) as opposted to entity-valued as listed in Table 4. We employ rules to extract these fillers. We build separate relation extractors for each attribute slot based on a set of hand-tuned rules. Each extractor takes an entity mention and the sentences in which it appears as input, and outputs a sequence of attribute relation instances. Extractors also provide a confidence score for each extracted relation that is hard-coded and determined empirically. These confidence scores are used in post-processing for determining which relations should be included in the final output.

Slot	Patterns
employeeOf	PER—pay by state of—LOC, pay by state of, who resign from, have resign in protest from, have be select as candidate of, PER—cartoonist at—UKN, be announce as be, LOC—NNP attend summit of—ORG have make his/her legend since sign from, make start in, work as register lobbyist for, PER—defender in NNP—ORG PER—executive who run—UKN, replace NNP NNP as face of, per:organizations_founded , PER—be play position for—ORG
cityOf Residence	be perform with NNP NNP at, PER—revenue NNP—PER, PER—have be from—ORG PER—be live in—LOC, NNPS on death row in, PER—citizen of—LOC PER—family live in—LOC, PER—his/her brother live in—LOC, be wheel into court in, PER—where X grow up—LOC, who be away from, be son of NNP NNP NNP of, PER—who be a native of—LOC X, a Y prisoner; bounce between his/her home in, since X live in Y
schools Attended	from his/her days in college at, be junior at, revel in NNP NNP success at hockey player for, be graduate of NNP NNP NNP, have be select as candidate of, have resign in protest from, per:employee_or_member_of director of NNP for NNPS at; PER—pay by state of—LOC; X, Y deputy managing director; PER—lawyer be hire by—ORG; policy adviser on NNP at PER—be coach at—ORG, PER—professor of education at—ORG, who earn at
spouse	into a career to, doled punishment, PER—X's ex-husband, Y—PER have file for divorce from her husband, file for divorce from, PER—X, Y's ex-husband—PER, PER—hubby—PER, marry actress, file for divorce from music producer, marry socialite; accompany by his wife; X, Y's ex-wife; X join her/his fiance Y, photographed with fiance, PER—X, Y's girlfriend—PER
subsidiaries	ORG—research growth rate for—LOC, its parent firm NNP NNP NNP, be subsidiary of, X, operated by Y:INV, UKN—populace be obliterate—ORG ORG—headquarters that NNP—ORG, UKN—court while—ORG, headquarters that NNP, sell NNP brokerage business, ORG—mecca—ORG ORG—X, a venture of Y:INV—ORG, ORG—X company Y—ORG, PER—be split into—ORG, ORG—which is bought by:INV—ORG ORG—which own percent of—ORG, ORG—NNP parent of—ORG ORG—parent company of—ORG, own NNP broadcast network
headquarters	ORG—lender base in—UKN, NNP announce at ORG—NNP NNP program in—LOC, ORG—be headquartered in—LOC ORG—think tank in—LOC, ORG—research center in—LOC ORG—X, a Y-based bank—LOC , ORG—X, Y organization—LOC ORG—policy group in—LOC, tenant right organization base in downtown

Table 2: Top similar patterns to the target slots.

Slot	Prec	Rec	F1
children	0.154	0.266	0.195
cities_of_residence	0.238	0.294	0.263
city_of_birth	0.083	0.077	0.080
city_of_death	0.478	0.314	0.379
countries_of_residence	0.149	0.204	0.172
employeeOf	0.435	0.118	0.186
origin	0.020	0.016	0.018
parents	0.023	0.103	0.038
schools_attended	0.063	0.034	0.044
siblings	0.025	0.231	0.044
spouse	0.138	0.271	0.183
state_of_death	0.188	0.120	0.146
states_of_residence	0.109	0.207	0.413
city_of_headquarters	0.125	0.125	0.125
country_of_headquarters	0.160	0.082	0.108
founded_by	0.025	0.100	0.040
member_of	0.007	0.250	0.013
parents	0.036	0.286	0.063
shareholders	0.037	0.412	0.068
members	0.010	0.045	0.017
state_of_headquarters	0.350	0.318	0.333
subsidiaries	0.053	0.235	0.086
top_employees	0.398	0.279	0.328
Overall	**0.094**	**0.183**	**0.125**

Table 3: **Entity-based Slots:** Performance on different slots. We first list person slots, followed by organization slots. Our approach obtains high recall.

Slot	Prec	Rec	F1
alternate_names	0.150	0.097	0.118
title	0.259	0.199	0.225
charges	0.130	0.058	0.080
date_of_birth	0.400	0.060	0.105
date_of_death	0.290	0.100	0.149
age	0.786	0.224	0.349
cause_of_death	0.417	0.182	0.253
religion	0.200	0.250	0.222
alternate_names	0.234	0.202	0.217
date_founded	0.556	0.313	0.400
Overall	**0.272**	**0.150**	**0.193**

Table 4: **String-based Slots:** Performance on attribute slots of people and organizations. We employ rule based extraction for these slots.

5.1 Trigger Based Attribute Relations

Many of our attribute relation extractors use a list of *trigger words* as signals. When an input sentence containing an entity mention contains a trigger word, the corresponding extractor attempts to extract an attribute relation instance involving the mention and the trigger word. This attempt involves considering features of the trigger/mention pair like: the number of words separating the two, the word sequence occurring between the mention and trigger, the order of the mention and trigger, etc. The trigger words for slots *per:cause_of_death*, *per:religion*, and *per:charges* are extracted from Freebase (similar to Roth et al. (2012)), while trigger words for other slots, usually consisting of small lists (less than 10) of words, are manually collected. Examples of trigger words for the *per:title* relation include: "secretary," "technician," "congressman," "rep.," etc.

We use the following rules for the extractors:

Title Extractor: Historically, titles are an extremely prominent relation in the TAC Slot Filling track. We designed extra rules for extracting titles. These rules are based on the dependency parse of the sentence containing the target mention. Specifically, if the parent or a sibling of the mention in the dependency tree is labeled as an "appositive," we attempt to extract a title relation between the token and the mention.

Date Extractor: We identify dates in each document in preprocessing using the Natty date parser (Stelmach, 2013). Extractors for date-related slots, such as *age*, *date_of_birth*, *date_founded*, etc., require both the presence of a trigger word and a date in the input sentence. Some date-related relation extractors may also extract relations of multiple types. For example, using the publishing date of a document in addition to an assertion of the age of a recently deceased person, the extractor of *age* would infer slots of *date_of_death*.

Alternate Names Extractor: Using the extracted knowledge base, we are able to directly extract *per:alternate_names* and *org:alternate_names* from the output of cross-document coreference.

Our results on 2013 are listed in Table 4. We observe that rule based extractors can achieve high precision, while sometimes suffer from lower recall. Further, we do not extract some relations that historically have been infrequent including *political_religious_affiliation*, *number_of_employees_members* and *org:website*.

6 Relation Resolution

In part due to our slot filling (relation extraction) strategy and in part due to some relation extraction not being performed jointly (e.g. attribute relations), the set of extracted fillers for a mention/slot pair can be invalid with respect to the slot. For example, our relation extraction system could extract inconsistent *per:date_of_birth* values for the same entity. Additionally, our system may extract the same relation multiple times. To remedy these problem, we employ a suite of simple *resolvers* that handle these issues on a case by case basis.

For single-valued slots (i.e. *per:age*), we use a resolver that simply outputs the filler with the highest confidence. For multiple-valued slots (i.e. *per:alternate_names*), we threshold the number of (unique) fillers output by only picking the top k–ranked either by confidence or by the number of appearances (i.e. the filler "50" was extracted for entity *e_1* and relations *per:age* 20 times). By thresholding, we hoped to increase precision without much decrease to recall.

7 Conclusion and Future Work

We presented Universal Schema, our overall framework for TACKBP slot filling and cold start. It differs from existing systems since it considers all entities, relations, surface patterns, and annotations jointly in a holistic manner. We effectively construct a database about the entities in all of the input documents (training and test), and then querying this database to provide our submission.

A number of avenues exist for future work. We feel our prediction suffered due to the use of a basic cross-document system, and a within-document coreference system does not utilize information from the cross-document coreference system; we look forward to incorporating sophisticated cross-document coreference (Singh et al., 2011) and KB-supervised within-document coreference system (Zheng et al., 2013). Future work will also explore expansion of the features used in the universal schema model, including lexicons, more Freebase relations, and entity types (Yao et al., 2013).

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval and in part by DARPA under agreement number FA8750-13-2-0020. 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

Eric Bengtson and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 294–303.

Marc-Allen Cartright, Samuel Huston, and H Field. 2012. Galago: A modular distributed processing and retrieval system. In *SIGIR 2012 Workshop on Open Source Information Retrieval*, pages 25–31.

Michael Collins, Sanjoy Dasgupta, and Robert E. Schapire. 2001. A generalization of principal component analysis to the exponential family. In *Proceedings of NIPS*.

Laura Dietz and Jeffrey Dalton. 2012. Across-document neighborhood expansion: Umass at tac kbp 2012 entity linking. In *Proceedings of the Text Analysis Conference (TAC)*.

Fang Liu and Jun Zhao. 2012. Sweat2012: Pattern based english slot filling system for knowledge base population at tac 2012. In *Proceedings of the Text Analysis Conference (TAC)*.

A. McCallum, K. Schultz, and S. Singh. 2009. Factorie: Probabilistic programming via imperatively defined factor graphs. In *Neural Information Processing Systems Conference (NIPS)*.

Bonan Min, Xiang Li, Ralph Grishman, and Ang Sun. 2012. New york university 2012 system for kbp slot filling. In *Proceedings of the Text Analysis Conference (TAC)*.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of NAACL*.

B. Roth, G. Chrupala, M. Wiegand, M. Singh, and D. Klakow. 2012. Generalizing from freebase and patterns using distant supervision for slot filling. In *Proceedings of the Text Analysis Conference (TAC)*.

Sameer Singh, Amarnag Subramanya, Fernando Pereira, and Andrew McCallum. 2011. Large-scale cross-document coreference using distributed inference and hierarchical models. In *Association for Computational Linguistics: Human Language Technologies (ACL HLT)*.

Joseph Stelmach. 2013. Natty, October.

Limin Yao, Sebastian Riedel, and Andrew McCallum. 2013. Universal schema for entity type prediction. In *Proceedings of the 3rd Workshop on Automated Knowledge Base Construction*, October.

Jiaping Zheng, Luke Vilnis, Sameer Singh, Jinho Choi, and Andrew McCallum. 2013. Dynamic knowledge-base alignment for coreference resolution. In *Conference on Computational Natural Language Learning (CoNLL)*.