Story-Level Inference and Gap Filling to Improve Machine Reading

Hans Chalupsky
USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292 U.S.A.
hans@isi.edu

Abstract

Machine reading aims at extracting formal knowledge representations from text to enable programs to execute some performance task, for example, diagnosis or answering complex queries stated in a formal representation language. Information extraction techniques are a natural starting point for machine reading, however, since they focus on explicit surface features at the phrase and sentence level, they generally miss information only stated implicitly. Moreover, the combination of multiple extraction results leads to error compounding which dramatically affects extraction quality for composite structures. To address these shortcomings, we present a new approach which aggregates locally extracted information into a larger story context and uses abductive constraint reasoning to generate the best story-level interpretation. We demonstrate that this approach significantly improves formal question answering performance on complex questions.

Introduction

Machine Reading (MR) (Strassel et al. 2010) aims at creating coherent formal representations of knowledge from reading natural language texts. While this has been a long-standing goal of the AI and NLP communities, there has been a recent resurgence of this topic with a focus on naturally occurring texts at potentially very large scale. The results of an MR system should enable a computer program to execute some performance task, for example, an inference engine answering complex queries specified in a formal language, or a fraud detection system spotting potential securities fraud from analyzing financial news and companies' annual reports. This is somewhat of a departure from tasks such as question answering, summarization, information extraction (IE) or translation where results are usually intended for human consumption only and do not have to adhere to the rigor and brittleness of a formal representation language and ontology. MR aims for a higher level of text understanding than IE by requiring results to be globally coherent with respect to a background theory and by exploiting inference both for extraction and use of results. In its most general form, machine reading encompasses all forms of knowledge, such as ontologies, rules and instance knowledge; however, here we focus on the generation of instance knowledge only.

Despite the difference in emphasis, a traditional IE engine is a natural starting point for an instance-level MR system. Such an engine typically performs POS-tagging, parsing, mention and named entity detection, coreference resolution and relation detection, and a baseline approach could simply map its entity types and relations onto the target ontology of interest. Suppose we are using this to build an MR system in the domain of American football where we are interested in creating detailed accounts of football games from reading sports news. The resulting knowledge base should contain game instances, teams playing, winners and losers, points scored, scoring events of interest, players scoring, their team associations, game location, home/away team, game date, etc. Now consider the following news story paragraph:

> San Francisco's Eric Davis intercepted a Steve Walsh pass on the next series to set up a seven-yard Young touchdown pass to Brent Jones. Doug Brien missed a conversion kick, but the 49ers still led 13-3 early in the second quarter.

The first problem immediately apparent are the large number of gaps, that is, things left implicit, since they can be inferred or are uninteresting in this context. For example, only Davis' team association is explicitly described, figuring out the teams of the four other players is left to the reader. Similarly, a football-literate reader can easily infer that Walsh and Young are (likely) the quarterbacks of their respective teams. Finally, we can infer that San Francisco and the 49ers must be the same team, since that is the only way one could arrive at a 13-3 score after a touchdown that counts for at least six points. Since current IE techniques generally focus on surface features at the phrase and sentence level, many of these unstated implicit relationships will remain undetected.

The second problem we are faced with is error compounding: given such a knowledge base of games extracted by an MR system, it becomes easy to ask complex conjunctive queries, for example, "find all losing teams of games where San Francisco was the game winner, scored at least two touchdowns and missed one conversion kick". An answer to such a query will be based on multiple relations extracted by the underlying IE engine and their extraction

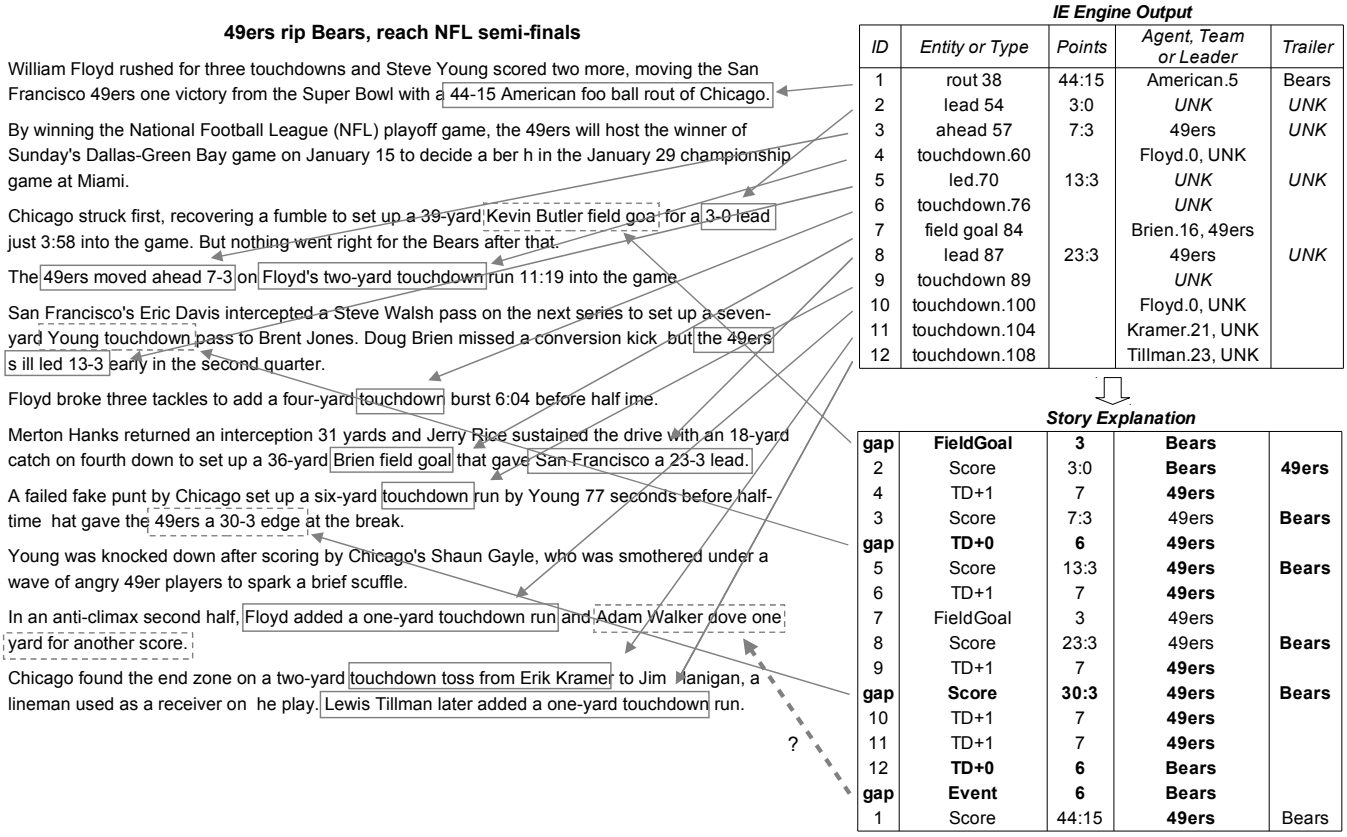

Figure 1: Example information flow from text to IE-engine outputs to detailed story explanation event sequence

errors will usually compound. For example, a state-of-the-art per-relation extraction f-score of 0.7 (cf. (Jiang and Zhai 2007)) will lead to a much lower per-query f-score on the order of $0.7^3 = 0.34$ for queries with three relations.

A key reason for both of these problems is that IE engines primarily focus on the sentence level and do not take the larger story structure nor any background knowledge into account. Human readers, on the other hand, build an evolving representation of the story they are told, where different portions complement and constrain each other, and where background knowledge helps to fill in story gaps.

In this paper we describe a story-level inference process we call *story explanation* to address these problems. Our approach aggregates locally extracted information from an IE engine into a larger story context and uses abductive constraint reasoning to generate the best story-level interpretation. We demonstrate that this approach significantly improves relation extraction and question answering performance on complex questions.

We study our approach in the domain of sports news such as reports about American football games. Game reports are interesting, since they often require a reading of the whole story to fit all the pieces together. The limited domain makes it more tractable to model relevant domain knowledge, yet the language is still highly variable and rich in difficult NLP problems such as metaphor, anaphora or complex coreference between aggregate event descriptions, to name just a few. Moreover, a large number of databases compile all kinds of information about sports teams and games, which facilitates gold standard creation and result evaluation.

Knowledge Aggregation

Figure 1 shows an example information flow in our machine reading system. We start with a natural language document such as the particular game report shown, and send that to a state-of-the-art information and relation extraction engine. The extraction engine we are using is a statistical extractor called SIRE (Florian et al. 2004), however, our approach is not tied to that particular technology. SIRE performs a variety of NLP tasks such as parsing, mention detection, entity detection/coreference resolution that groups mentions into equivalence classes, and relation detection. The outputs of SIRE that form the primary inputs to our machine reading system called *Knowledge Aggregator* are entities and their types, e.g., "William Floyd" of type "Person", and relations between entities, for example, "William Floyd" "agentOf" "touchdown". SIRE is a supervised system that uses annotated documents to learn a statistical extraction model specific for a domain. In this case, SIRE was trained to extract entity types and relations specific to football games in addition to a set of generic entity and relation types.

The boxes around words and phrases in the story in Figure 1 indicate a subset of the entities and relations SIRE detects in this document. A more detailed account of these is given in the table labeled "IE Engine Output", where each

row represents a brief summary of relevant information extracted for an event or score snapshot. The arrow points to the text portion where this information is coming from. Each row is identified by an ID (not generated by SIRE) used to identify it with the corresponding row in the story explanation below. The second column lists specific event and score entities extracted by SIRE. All of these also have generic types associated such as "EventPlayScored" or "EventOutcomeFinal". These types are not shown in the table but considered by the story explanation system.

The next two columns describe what we know, based on IE engine output only, about the agent of a particular scoring event (person or team) or who the leading/trailing teams are for a score snapshot. Note that for many rows that information is fully or partially unknown, e.g., for Row 2, or incorrect as for the leading team in Row 1. These gaps are either due to information left implicit in the story, long distance relationships as in the third paragraph, or extraction errors that link incorrect information or miss relevant information such as the field goal by Kevin Butler. If we base the extraction of an MR target relation such as "gameWinner" on IE output alone, we will often get no or an incorrect result such as "American".

In the story explanation step (described in more detail in the next section), the system uses abductive constraint reasoning to fit all this information together, correct likely errors and fill in missing information to come up with the most plausible event sequence that explains the data extracted by IE. The table labeled "Story Explanation" shows the result of this process. Each item or row shown in bold font shows an addition, elaboration or correction of an IE datum. The sequencing of rows in this explanation is significant. Rows labeled "gap" explain scoring events or scores that were missed by IE (indicated in text by the dotted boxes). After this explanation step, we now know the proper winner of this game. However, not all of the explanation details are correct. While the system correctly predicts the last two scoring events to be by the Bears with six points each, the touchdown in Row 11 by Kramer is attributed to the 49ers, which is incorrect. Nevertheless, if we only look at the team level, all the information in the explanation is correct.

In the final step not shown in the example, the Knowledge Aggregator uses the generated story structure to populate a knowledge base with instances and relations from a particular target ontology. For example, we can now directly read off game winner, loser, points scored, subevent structure, detailed event types, player-team associations, numbers of scoring events by type, etc. Moreover, the subevent structure can help with the determination of other information such as game date, location or whether the game went into overtime, since often pertinent information will be linked to a subevent instead of the game event itself. This last step is similar to the template generation step in a traditional information extraction system (Hobbs and Riloff 2010), with the exception that the result instantiates some arbitrary target ontology which might be significantly removed from the structures expressed directly in language.

Story Explanation

Conceptually, story explanation can be separated into the following phases:

1. Story classification
2. Story segmentation
3. Skeleton construction
4. Hypothesis generation
5. Abductive explanation
6. Explanation evaluation

Our approach is implemented using a general logical inference framework, hence, these phases will be automatically interleaved by the underlying inference engine.

Story classification determines whether a particular document is amenable to explanation. In the current system, explanation is only attempted in documents deemed to be reports on a single game, since multi-game stories pose a more difficult segmentation challenge. To determine the document class, the system uses a simple heuristic based on the number of final scoring events reported by the IE engine.

Segmentation finds entities and events of interest, for example, scoring events and intermediate and final score reports, and collects them into a set of story objects that will be subject to explanation. Aggregate entities, e.g., "Floyd rushed for three touchdowns" are currently excluded from this step, since they are difficult to coreference with other individually reported events.

Skeleton construction imposes a likely temporal ordering on the scores and events of interest. For example, in Figure 1, the final score was reported first, followed by some more detailed description in the body of the story. We use a simple heuristic that orders scores by their points and assumes temporal sequencing of events from presentation order (allowing for some minor proximity-based variations such as a scoring event closely following the score it caused). Future versions will also take temporal qualifications such as "two minutes into the second half" into account. Constraining event order is important, otherwise, the space of possible event sequences quickly becomes intractable.

Hypothesis generation produces candidate bindings for certain variables in the story structure. For example, in order to properly account for a scoring event, we need to know which team was responsible for it. If a team can be inferred directly or from a player whose team is known, or from a team association to an event that caused the scoring event, that team will be used (e.g., for Row 7 in Figure 1). If no such team can be found, possible team candidates from the rest of the document will be used. Moreover, even if a team is found it might be incorrect, hence, we always have to supplement with a set of additional possibilities. The predicates "possiblePlayTeamsOf" and "possiblePlayScores" in the example rule in Figure 2 are implementing this type of hypothesis generation (the latter for types of plays and associated points, e.g., a touchdown might count for 6,7 or 8 points). The most likely candidates are always generated first in order to produce simple, plausible explanations quickly.

Abductive explanation then combines domain knowledge and hypothesis generation to fit a set of story elements into a plausible picture. Due to space constraints, we can only

```
(=> (and (lastElement ?story ?play)                                  ....explain ?story starting from its end
         (ScoringPlay ?play)                                         ...?play is a scoring event
         (nthHead ?story -1 ?remainder)                              ...?story prefix ?remainder prior to ?play
         (gameExplanation ?subt1 ?subt1Pts ?subt2 ?subt2Pts ?remainder ?all ?subEvSeq)   ...try to explain ?remainder
         (possiblePlayTeamsOf ?play ?all ?playTeams)                 ....?playTeams hypotheses for ?play
         (or (memberOf ?playTeam ?playTeams)
             (and (= (cardinality ?playTeams) 0)                     ....if we don't know any possible agents for
                  (memberOf ?playTeam (listof ?subt1 ?subt2))))      ....?play, try teams of subexplanation
         (memberOf (listof ?playType ?playPts) (possiblePlayScores ?play))   ....?playType hypotheses for ?play
         (or (and (= ?subt1 ?playTeam)                               ...leading team ?subt1 scored with ?play
                  (= ?t1 ?subt1) (= ?t2 ?subt2)
                  (+ ?subt1Pts ?playPts ?t1Pts)
                  (= ?t2Pts ?subt2Pts))
             (and (= ?subt2 ?playTeam)                               ...trailing team ?subt2 scored with ?play
                  (+ ?subt2Pts ?playPts ?newPts)
                  (or (and (> ?newPts ?subt1Pts)                     ...lead changed
                           (= ?t1 ?subt2) (= ?t2 ?subt1)
                           (= ?t1Pts ?newPts) (= ?t2Pts ?subt1Pts))
                      (and (=< ?newPts ?subt1Pts)
                           (= ?t1 ?subt1) (= ?t2 ?subt2)
                           (= ?t1Pts ?subt1Pts) (= ?t2Pts ?newPts)))))
         (insertLast ?subEvSeq (eventDetail ?play ?playType ?playTeam ?playPts) ?evSeq))  ....add event to explanation
    (gameExplanation ?t1 ?t1Pts ?t2 ?t2Pts ?story ?all ?evSeq))
```

Figure 2: Example game story explanation rule

show one of the story explanation rules involved in Figure 2. This rule (written in KIF) handles the case where the current element is a scoring event such as a touchdown. It then enumerates possible types and agents for this event, connects those with different possible explanations of the remaining prior events, and computes a score tally and detailed event description that then become part of one possible story explanation. In case there are not enough candidate events to explain known score snapshots, unknown events are hypothesized (e.g., the initial field goal event in Figure 1). If the system is overconstrained, events will be ignored to relax some constraints. Even though this is an abductive inference process, the use of hypothesis generators such as "possiblePlayScores" allows us to do this with a deductive prover.

Finally, multiple competing explanations are evaluated based on a simple cost model for ignored and hypothesized events. Based on these evaluations, the least-cost explanation that covers the maximum number of game entities and relations while still obeying all constraints is chosen to populate the resulting knowledge base.

The underlying rule base contains about 110 domain-specific and 30 domain-independent rules to support interaction with the IE system, text-level and domain-specific inference and story explanation. It took about one man month for a skilled knowledge engineer to develop. This is comparable to the document annotation time for about 80 documents to train the extraction engine. We use PowerLoom as our representation and inference engine which can analyze a corpus of 500 news stories in about 30 minutes on a standard Intel i7 desktop exploiting eight-core parallelism.

Experimental Setup

To evaluate our story explanation approach, we use formal question answering as a performance task. The system gets as input a set of natural language documents and a set of queries expressed in RDF. The types and relations in each query are restricted to those in the given target ontology. To answer a query the system has to map it onto the knowledge extracted from the set of corpus documents and use some limited inference to handle type and relation subsumption plus a small set of domain-specific rules. The output of the system is a set of RDF graphs where each graph instantiates one of the queries by providing bindings for its variables. We measure performance by comparing the resulting graphs to the ones generated by running the queries over a gold-standard corpus created by human annotators. This comparison yields standard recall, precision and f-measures based on the number of correct and incorrect graphs produced by the system. To evaluate the impact of story explanation, we compare query results against a baseline version that maps IE results directly onto the target ontology without any story explanation inference.

For our experiments, we used a corpus and target ontology in the domain of games in the American National Football League (NFL). The ontology describes types of games, NFL teams, game winners, losers, points scored, numbers of game periods, overtime, numbers of scoring events, etc. and is described in some more detail in (Strassel et al. 2010). The corpus LDC2009E112 V1.2 was developed by the Linguistic Data Consortium as a training corpus for the DARPA Machine Reading Program. It contains 110 news stories (from NYT, AFP & APW) with about 600 words on average describing (mostly) football-related content. A gold standard was created by LDC annotators who annotated the corpus for concepts and relations in the target ontology that were observable or inferable more or less directly from text.

A typical content sentence would be the following: "Denver again proved inhospitable to the Kansas City Chiefs, as the Broncos pounded them 30-10 in a National Football League Monday night game." From this sentence an annota-

Query ID	English Paraphrase
NFLGame-date	Find all games on *date* where *team1 played* team2
NFLGame-scores	Find all games where *winner* played *loser* and their respective *points* scored
NFLGame-2TDs+	Find all games where *loser* scored *touchdowns* and *touchdowns* >= 2
NFLGame-OT-win	Find all games where *winner* won the game in overtime
Packers-losers	Find all games where *loser* lost to the Green Bay Packers and the loser's *points* scored

Table 1: Test query IDs and their English paraphrases. Note that queries are formulated in RDF and that the underlined words above correspond to output variables in a SPARQL select statement.

Query ID	Gold Standard Answers	IE Baseline					with Story Explanation					F1 Change
		TPs	FPs	R	P	F1	TPs	FPs	R	P	F1	
NFLGame-date	232	29	74	0.125	0.282	0.173	44	92	0.190	0.324	**0.239**	**38.1%**
NFLGame-scores	155	52	12	0.335	0.813	0.475	64	22	0.413	0.744	**0.531**	**11.8%**
NFLGame-2TDs+	44 (51)	27	1	0.529	0.964	0.684	31	4	0.608	0.886	**0.721**	**4.6% (5.5%)**
NFLGame-OT-win	10	0	0	0.000	0.000	0.000	1	0	0.100	1.000	**0.182**	∞
Packers-losers	7	4	1	0.571	0.800	0.667	4	1	0.571	0.800	0.667	0.0%

Table 2: Story explanation improvement over IE baseline on several test queries

tor might infer that Denver played Kansas City, Denver was the home team and won with a final score of 30, Kansas City lost with a score of 10, and that the game was on a Monday night which in conjunction with the article date would allow us to infer the actual game date.

A typical evaluation query in this domain looks like this (in SPARQL, namespace prefixes omitted):

```
SELECT ?Team_L ?Points_L ?TD_L WHERE {
    ?G type NFLGame .
    ?G gameLoser ?Team_L .
    ?G teamFinalScoreInGame ?SS_L .
 ?SS_L type NFLGameTeamSummaryScore .
 ?SS_L teamScoringAll ?Team_L .
 ?SS_L pointsScored ?Points_L .
 ?SS_L touchdownCompleteCount ?TD_L .
 FILTER(?TD_L >= 2) }
```

This query corresponds to query "NFLGame-2TDs+" in Tables 1 and 2. An answer graph instantiating this query counts as correct if a matching graph exists in the ground truth. For a match to count, only nodes instantiating query output variables need to match (e.g., ?Team_L, ?Points_L, ?TD_L), internal variables such as the game event ?G or score object ?SS_L are treated as anonymous skolems. Duplicate correct answers that might come from different stories reporting on the same game are counted only once. Missing any aspect of the answer graph will render the answer incorrect, there is no partial credit.

Experimental Results

To evaluate our system, we first run it in a baseline configuration without explanation, and then with story explanation inference turned on. For the baseline, game winners and losers reported by the IE engine are simply linked together and taken at face value (we use the IE engine's highest confidence results only). All other inference having to do with game dates, locations, home/away teams, overtime, game types, etc. is identical to the full version.

With story explanation turned on, the system tries to determine first whether a document is a single or multi-game report. Only if a document is likely a single-game story will it attempt to do full story explanation, since, due to the current lack of a good document segmentation approach, story explanation gets confused on multi-game documents. For multi-game reports, it will behave identical to the baseline version. In our test corpus, 44 out of the 110 documents are determined by the system to be single game stories, and 33 as multi game stories with about three games on average (but a maximum of 12). This classification is of course noisy, since it is based on IE engine output.

English paraphrases for five test queries presented to the system in RDF are shown in Table 1. Evaluations of query results are summarized in Table 2. For four out of the five queries the story explanation system produces a significant improvement in query f-score. The most representative query in the evaluation set for our purposes is NFLGame-scores, since (1) it has a large enough number of answers in the test corpus to be statistically meaningful, and (2) its results can be read off directly from a successful game story explanation. Moreover, while the IE engine is fairly good in finding game outcomes and their associated teams, it often gets mixed up about the respective winning and losing teams due to the highly variable and metaphorical language used to describe game outcomes (e.g., "Denver again proved inhospitable to the Chiefs, as the Broncos pounded them 30-10..."). In those cases additional constraints from plays described in the story can be exploited by the story explanation mechanism to correct IE errors or omissions.

Queries such as NFLGame-date and NFLGame-OT-win benefit from story explanation due to the additional opportunities to link up game time or date specifications. For example, in the one positive result for NFLGame-OT-win, story explanation connects the following field goal to the final game outcome: "Vinatieri found enough solid footing to kick a 24-yard field goal with 19 seconds left in overtime". Since IE extracted the appropriate temporal link connecting the field goal to a temporal expression signifying overtime, it can infer that the corresponding game outcome (linked by story explanation) occurred in overtime as well.

For the NFLGame-2TDs query, the gold standard annotations were very conservative, since annotators were not allowed to make inferences about touchdowns scored and only annotated explicit mentions such as "won by two touchdowns." This led to a disproportionate number of false positives reported, since both systems would make such inferences when warranted from intermediate events or final scores (e.g., a score of 24 has a 0.905 probability to include at least two touchdowns). To compensate, we manually inspected false positives and supporting provenance and reclassified them as true where warranted. The improvement of 4.6% is based on a revised gold standard including now 44 true answers. A more accurate estimate of true answers is 51 (based on a 70% increase in true positives after inspection), and a revised improvement of 5.5% is reported in parentheses. The other queries did not rely on inferences by annotators and did not need any gold standard corrections.

These improvements are especially significant, since only about 20-30% of all games are reported on in single-game documents. For the rest, no story explanation is attempted. In general, recall improves with some degradation of precision possible. This is to be expected given the somewhat simplistic heuristics for story segmentation and event reordering.

Related Work

How to use knowledge and inference to facilitate language understanding has been researched since the early days of AI (e.g., (Schank 1975)). Similarly, that abduction plays an important role in interpretation has been clear at least since Hobbs' seminal paper (1993). Only more recently, however, have these approaches been evaluated on naturally occurring texts at scale. For example, the work of Harabagiu et al. (2003) on answer mining and Ovchinnikova et al. (2011) on textual entailment. In these works, abduction is only used at the sentence level, while our system performs abductive interpretation across a whole story, which requires different segmentation and reasoning techniques to make it tractable. Moreover, our system identifies and closes gaps (implicit facts) which can then be queried, while a QA system such as Harabagiu's only identifies answers explicitly mentioned in text. Roth and Yih (2007) formulate a linear programming approach for global inference and constraint reasoning. However, they only apply it to individual entity and relation detection tasks as opposed to assembling whole story explanations. Our approach would benefit from more sophisticated document classification and segmentation techniques as developed in the IR community (e.g., (Sun et al. 2007)) which would allow us to apply explanation to a larger portion of documents. Moreover, it requires a significant amount of manually coded domain knowledge, and we are currently investigating how approaches such as (Chambers and Jurafsky 2009) could be used to learn some of the requisite knowledge automatically.

Conclusion

We presented a new abductive story-level inference approach to support machine reading applications. Our approach generates the best story-level interpretation for the entities and relations generated by a state-of-the art information extraction engine based on background knowledge and a set of story explanation rules. This process can fill gaps of missing or implicit information and counteracts error compounding for queries that involve multiple relations. We applied our approach to a domain of sports news articles and demonstrated that it significantly improves relation extraction and question answering performance on a set of complex questions in this domain.

Acknowledgements

This research was supported by the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) contract no. FA8750-09-C-0172. Any opinions, findings, conclusion or recommendations expressed in this material are those of the author and do not necessarily reflect the view of DARPA, AFRL, or the US government.

References

Chambers, N., and Jurafsky, D. 2009. Unsupervised learning of narrative schemas and their participants. In *ACL '09*.

Florian, R.; Hassan, H.; Ittycheriah, A.; Jing, H.; Kambhatla, N.; Luo, X.; Nicolov, N.; and Roukos, S. 2004. A statistical model for multilingual entity detection and tracking. In *NAACL/HLT*, 1–8.

Harabagiu, S.; Moldovan, D.; Clark, C.; Bowden, M.; Williams, J.; and Bensley, J. 2003. Answer mining by combining extraction techniques with abductive reasoning. In *TREC*, 375–382.

Hobbs, J., and Riloff, E. 2010. Information Extraction. *Handbook of Natural Language Processing*.

Hobbs, J.; Stickel, M.; Appelt, D.; and Martin, P. 1993. Interpretation as abduction. *Art. Int.* 63(1–2):69–142.

Jiang, J., and Zhai, C. 2007. A systematic exploration of the feature space for relation extraction. In *NAACL/HLT*, 113–120.

Ovchinnikova, E.; Montazeri, N.; Alexandrov, T.; Hobbs, J.; McCord, M.; and Mulkar-Mehta, R. 2011. Abductive reasoning with a large knowledge base for discourse processing. In *Proceedings of the 9th International Conference on Computational Semantics (IWCS'11)*, 225–234.

Roth, D., and Yih, W. 2007. Global inference for entity and relation identification via a linear programming formulation. In Getoor, L., and Taskar, B., eds., *Introduction to Statistical Relational Learning*. MIT Press.

Schank, R. 1975. Using knowledge to understand. In *Proceedings of TINLAP '75*, 117–121.

Strassel, S.; Adams, D.; Goldberg, H.; Herr, J.; Keesing, R.; Oblinger, D.; Simpson, H.; Schrag, R.; and Wright, J. 2010. The DARPA Machine Reading Program-Encouraging Linguistic and Reasoning Research with a Series of Reading Tasks. *Proceedings of LREC 2010*.

Sun, B.; Mitra, P.; Giles, C.; Yen, J.; and Zha, H. 2007. Topic segmentation with shared topic detection and alignment of multiple documents. In *SIGIR '07*, 199–206.