Unsupervised Multi-Lingual Cold Start Slot Filler Ensembling with the Knowledge Resolver System for TAC-KBP 2016

Hans Chalupsky
USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA, USA
hans@isi.edu

Abstract

This paper describes a new slot filler ensembling component we developed as part of our Knowledge Resolver relation extraction and knowledge base population toolkit (KRes), and its performance on the TAC-KBP 2016 ensembling task. At the core of our approach is a process of *anti-error compounding* which identifies likely redundant extractions that are based on at least partially independent evidence, and aggregates them into more reliable results exploiting the redundancies. We use mention overlap, string matching and within and cross-document coreference provided by an EDL system to determine likely equivalent extractions. We also exploit relative system performance estimates based on preliminary scoring or past performance where available. Our system does not require any training. By merging the top-5 recall KBs created by different teams based on preliminary score estimates distributed with the TAC-KBP 2016 SFV evaluation package, our system improved the best overall English SF-All-Micro f1 score for Hop-0 queries from 0.319 to 0.350, and for Hop-0+1 queries from 0.270 to 0.286. For Spanish and Chinese we could not create any improvements due to a lack of run diversity for Spanish and a token-based name and value matcher that was not working sufficiently for Chinese.

1 Introduction

This paper describes a new slot filler ensembling component we developed as part of our Knowledge Resolver relation extraction and knowledge base population toolkit (KRes) (Chalupsky, 2012; Chalupsky, 2013; Chalupsky, 2014; Chalupsky, 2015), and its performance on the TAC-KBP 2016 ensembling task. At the core of our approach is a process of *anti-error compounding* which identifies likely redundant extractions that are based on at least partially independent evidence, and aggregates them into more reliable results exploiting the redundancies. We use mention overlap, string matching and within and cross-document coreference provided by an EDL system to determine likely equivalent extractions. We also exploit relative system performance estimates based on preliminary scoring or past performance where available. Our system does not require any training. By merging the top-5 recall KBs created by different teams based on preliminary score estimates distributed with the TAC-KBP 2016 Slot Filler Validation (SFV) evaluation package, our system improved the best overall English SF-All-Micro f1 score for Hop-0 queries from 0.319 to 0.350, and for Hop-0+1 queries from 0.270 to 0.286. For Spanish and Chinese we could not create any improvements due to a lack of run diversity for Spanish and a token-based name and value matcher that was not working sufficiently for Chinese.

2 Approach

2.1 Anti-Error Compounding

At the core of our approach is a process of *anti-error compounding* which identifies likely redundant extractions that are based on at least partially independent evidence, and aggregates them into more reliable results exploiting the redundancies. The intuition behind our approach is as follows. Suppose we have two independent relation extractions such as these:

```
P1 per:employee_or_member_of O1,  c1
P2 per:employee_or_member_of O2,  c2
```

c_1 and c_2 are probabilities of correctness (or confidence scores) for the particular relation. Suppose we have additional evidence that O_1 equals O_2 with probability c_3, then we can chain the two relations to conclude that P_1 and P_2 work in the same organization. The probability of correctness of this chained result is $c_1 * c_2 * c_3$ (assuming these probabilities are independent), leading to the well-known phenomenon of error-compounding when combining multiple noisy extractions, since now the error probability is $1 - c_i^n$ which increases rapidly with increasing n. For example, combining three extractions with $c_i = 0.7$ leads to an error probability of 0.66 for the combined result.

Error compounding occurs for the probability that *all* of the chained relations are correct. However, the probability that *at least one* of two relation extractions is correct is $1 - (1 - c_1) * (1 - c_2)$, that is that they are not both incorrect, which is the anti-probability of the product of the error probabilities, therefore we call this *anti-error compounding*.

Why is this useful? In the above example we know the probability of at least one extraction being correct, but not which one. Suppose we have (perfect for now) evidence as before that O_1 equals O_2 and now also that P_1 equals P_2, thus, now both extractions do in fact express the same relation. If we again assume independent individual extraction probabilities of 0.7, the probability that at least one of them is correct is now 0.9, and, since they are both the same, we now know which one (both of them), and we now magically boosted our extraction probability of correctness. Another matching relation would boost this further to 0.97, which shows how the error now exponentially decreases in the same way it increased before.

Unfortunately, of course, reality is much more complex. Evidence of equality between relation arguments is itself associated with noise generally requiring some form of coreference resolution, and that error probability multiplies into the boosted extraction probability from above. Nevertheless, there are cases where we can be highly confident of argument equality, for example, if arguments come from the same or highly overlapping text spans in a document. In such cases we will still get boosted relation confidence even though somewhat moderated by coreference probabilities.

Another problem is that probabilities of correctness are either not available or imperfect estimates, making it difficult to assess the correctness of an ensembled result. For example, rule-based or clustering-based coreference algorithms often do not provide confidence estimates. When confidences are available, they are estimated based on training data which makes them unreliable on test data that might be highly different from what was encountered in training.

Finally, relation extraction confidences and errors are generally not independent in the sense we assumed above. Redundant extractions might come from the same extractors and/or similar language and/or use similar tools or preprocessing, etc., therefore, the compounding effect from combining them will be less pronounced or possibly even worse than the individual correctness probabilities. For these various reasons we simply strive for maximizing independent pieces of evidence for particular slot fillers, but we do not attempt to calculate a more precise probability of correctness of an ensembled result to use as the basis for our ensembling decisions.

2.2 Slot Value Ensembling

Figure 1 shows the overall architecture of our 2016 TAC-KBP ensembling system. It exploits anti-error compounding to ensemble a number of different Cold Start Slot-Filling runs as follows:

(1) Run selection: in this stage we select a number of most promising input runs either based on a team or component's past performance and/or performance estimates based on a small manually evaluated sample. This part is not automatically supported by our approach, we simply select a fixed number of high-performing runs based on these performance indicators. Only a relative ranking of runs is required, we do not need or make use of absolute performance metrics. Since we are looking for maximally independent extractions, we combine runs from all different teams. Moreover, since our ap-

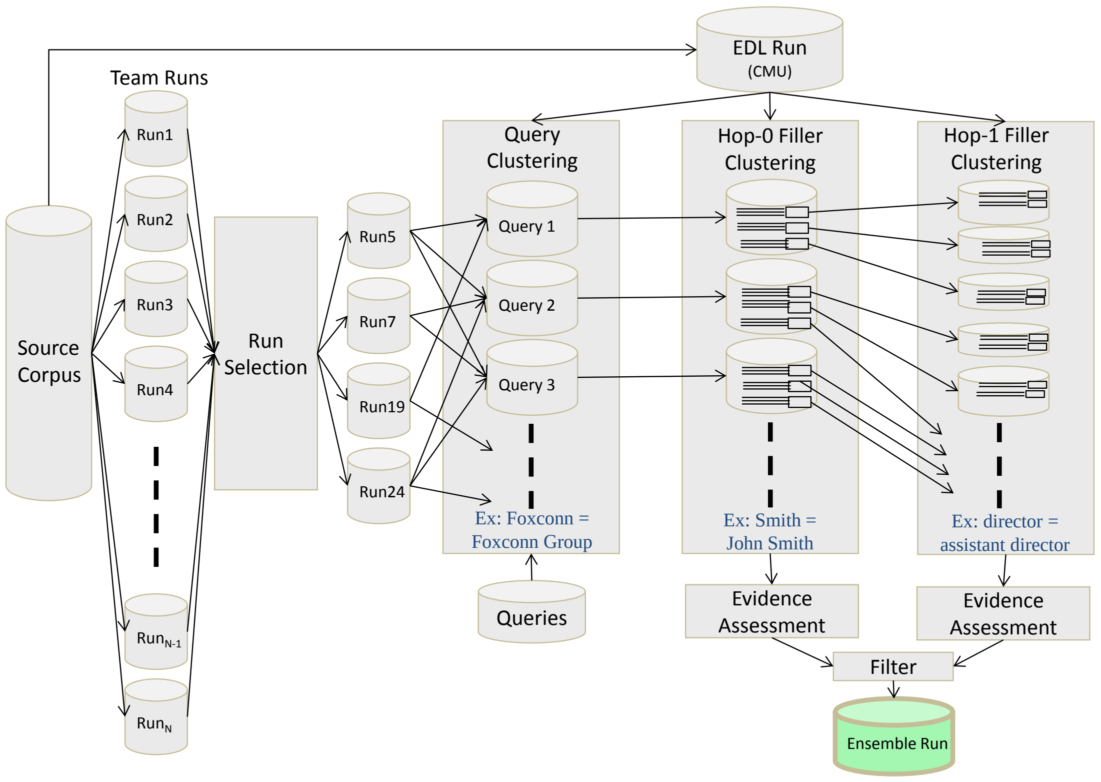

Figure 1: System architecture

proach requires multiple pieces of evidence for each ensembled extraction, we look for runs with maximum recall to increase the chance of multiple independent extractions. For the 2016 evaluation we picked 3, 5 or 10 runs from different teams ranked by preliminary recall estimates provided by linguists of the Linguistic Data Consortium (LDC) based on a manual assessment of a small sample of queries (for English 85 out of 1351 queries (6.3%) were in this sample assessed by LDC).

(2) Query and filler clustering: here we cluster plausibly equivalent relation arguments starting with query entities, leading to Hop-0 slot fillers and further leading from Hop-0 filler clusters to Hop-1 filler clusters. This process exploits a number of indicators such as text span identity and overlap, partial match between normalized token sequences (prefix, postfix, and permuted match) as well as within and cross-document coreference information provided by an entity linking and discovery (EDL) system. Note that this equivalence clustering is easier than solving a full within or cross document coreference problem, since we are constrained to the slot fillers starting from a specific query entity as opposed to all entities in a document or corpus. For example, assuming "director" and "associate director" as equivalent fillers for the `per:title` slot of "John Smith" is highly plausible, while the same might not be true in general. Moreover, over-merging at this stage is not too problematic, since that only provides possibly unwarranted additional evidence which will generally hurt precision but can also improve recall.

(3) Evidence assessment: given the above clustering, we have a set of plausibly equivalent Hop-0 and Hop-1 query results. For each cluster of equivalent fillers, we report a representative if we have at least two independent pieces of evidence. This step is somewhat complicated by the fact, that for Hop-1 queries, we have to ensure that their Hop-0 roots are equally multiply supported. Currently, we look for equivalent extractions that either come from two separate teams or from two separate documents. Runs from different teams increases independence

of errors due to different failure modes, however, many systems use standard tools for parsing, NER, etc., decreasing that independence. The intuition for looking for extractions from separate documents is that those are more likely to be based on different textual evidence (even though we are currently not explicitly testing for that).

(4) Filtering: finally, we perform some filtering to satisfy output constraints such as single-valuedness, which might have been violated by the merging process.

In its current form, this approach requires a sufficient number of independent input systems that cover the space of sought slot values reasonably well in multiple ways. If we have unbalanced coverage as occurred for the Spanish language condition, where we had a single team with a much higher score and coverage than the two other submissions, we will get a much smaller number of mergable extractions leading to low ensemble recall. This problem can be remedied with somewhat different merging strategies, however, currently this is not addressed.

3 Evaluation Results

We submitted a total of eight ensembling runs, four for English, two for Spanish and two for Chinese. All our development was done on English-only data from the 2015 TAC-KBP evaluation. No language-specific code was added for either Spanish or Chinese. Our best English ensemble run outperformed the best individual system run by 1.6 f1 points on the SF-All-Micro score (we use SF-All-Micro scores for all our comparisons below). Our best Spanish ensemble run scored better than two of the three teams that submitted runs, but significantly worse than the best-performing run. f1 dropped from 15% to 4%, while precision increased from 14.6% to 23.9%. For Chinese, our English-centric name matching really did not work giving us an f1 of 7% at the bottom of all individual team submissions. We discuss these results in more detail below.

Table 1 summarizes scores from our four submitted English SFV ensemble runs SAFT_ISI_1 to 4, a number of best team runs A to F based on various scoring aspects shown in the Run Characteristics column, and a number of post-evaluation experiments SAFT_ISI 9 to 16 we performed after we received the final scores and ground truth from LDC.

Each SAFT_ISI run uses a number of Cold Start SF and KB runs as inputs which were selected based on preliminary scores distributed at the beginning of the SFV evaluation window. These preliminary scores were generated from LDC assessments of 85 English evaluation queries out of 1351 total queries, of which 402 were fully assessed at the end of the evaluation. That is, preliminary scores were based on about 20% of the full ground truth that became available after the evaluation, which in itself is only a partial assessment of all query outputs. We only used preliminary scores for relative ranking of submitted runs, we did not make use of the detailed slot-value assessments in any way on which the preliminary scores were based.

Our best-performing Run 1 used the top-5 runs from all different teams based on Hop-0+1 recall as input. These runs vary greatly in overall performance with preliminary Hop-0+1 f1 values ranging from 8.7% to 23.1% and final values ranging from 11% to 26.9%. Our ensemble run achieved 28.64% Hop-0+1 f1 which improves upon the best-performing Team Run A by about 1.5 points. Later analysis revealed that our input runs were in fact selected based on their Hop-0 recall only (thus the asterisk in the table). We corrected that in post-eval Run 9 which gives us an additional minor improvement to 28.7%. Input runs for our Run 2 submission were based on preliminary Hop-0 recall only, however, due to the mistake these were identical to those from Run 1 leading to identical results. Run 3 merges inputs from the top-10 runs from all different teams based on Hop-0+1 recall (this time correctly so). These runs vary even more, with final f1 values ranging from 3.7% to 26.9%. The ensemble run has the same f1 as the best-performing Team Run A, but it achieves approximately 9 points higher recall and about 5 points higher recall than the highest recall Team Run B. For Runs 1 to 3, we used EDL results from our collaborators at CMU. Run 4 is identical to Run 1 in its input selection, however, it uses EDL results from a team that had a top-scoring submission in 2015. The overall f1 is identical to Run 1, so using the different EDL results did not make a difference. Run 4 suffered the same input run selection mistake as Run 1 which was corrected in post-eval Run 10 - again with identical results.

		Hop-0			Hop-1			Hop0+1		
SFV Run	**Run Characteristics**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
SAFT_ISI Run 1	top-5 prelim. hop0+1 recall (*)	0.3952	0.3146	**0.3503**	0.1849	0.1290	0.1520	0.3308	0.2525	**0.2864**
SAFT_ISI Run 2	top-5 prelim. hop0 recall (= to Run-1)	0.3952	0.3146	0.3503	0.1849	0.1290	0.1520	0.3308	0.2525	0.2864
SAFT_ISI Run 3	top-10 prelim. hop0+1 recall	0.2677	**0.3752**	0.3124	0.1753	0.1496	0.1614	0.2460	**0.2996**	0.2702
SAFT_ISI Run 4	top-5 prelim. hop0+1 recall (*) w/ EDL2	0.3952	0.3146	0.3503	0.1849	0.1290	0.1520	0.3308	0.2525	0.2864
CS SF/KB Run										
Team Run A	best final hop0+1 f1	0.4609	0.2437	0.3188	**0.2528**	0.1320	0.1734	0.3918	0.2063	0.2703
Team Run B	best final hop0+1 recall	0.2903	0.2792	0.2846	0.0524	**0.1877**	0.0819	0.1351	0.2485	0.1751
Team Run C	best final hop1 f1	**0.4657**	0.2408	0.3174	0.2542	0.1320	**0.1737**	**0.3947**	0.2043	0.2693
Team Run D	best final hop1 recall	0.2903	0.2792	0.2846	0.0524	0.1877	0.0819	0.1351	0.2485	0.1751
Team Run E	best final hop0 f1	0.4609	0.2437	0.3188	0.2528	0.1320	0.1734	0.3918	0.2063	0.2703
Team Run F	best final hop0 recall	0.2903	0.2792	0.2846	0.0524	0.1877	0.0819	0.1351	0.2485	0.1751
Post-Eval Run										
SAFT_ISI Run 9*	top-5 prelim. hop0+1 recall (* Run-1)	0.3943	0.3087	**0.3463**	0.2047	0.1290	0.1583	0.3396	0.2485	**0.2870**
SAFT_ISI Run 10*	top-5 prelim. hop0+1 recall w/ EDL2 (* Run-4)	0.3943	0.3087	0.3463	0.2047	0.1290	0.1583	0.3396	0.2485	0.2870
SAFT_ISI Run 11	*top-5 final hop0+1 recall*	*0.4133*	*0.3309*	***0.3675***	*0.2440*	*0.1496*	***0.1855***	*0.3662*	*0.2701*	***0.3109***
SAFT_ISI Run 12	top-5 prelim. hop0+1 f1	0.4057	0.2925	0.3399	0.1880	0.1378	0.1591	0.3320	0.2407	0.2790
SAFT_ISI Run 13	*top-5 final hop0+1 f1*	***0.5152***	*0.2747*	*0.3584*	***0.2815***	*0.1114*	*0.1597*	***0.4516***	*0.2200*	*0.2959*
SAFT_ISI Run 14	5 submitted KBs from Team Run A team	0.4305	0.2378	0.3064	0.2133	0.1408	0.1696	0.3489	0.2053	0.2585
SAFT_ISI Run 15	top-5 prelim. hop0+1 recall, diff. team only	0.4268	0.3058	**0.3563**	0.2471	0.1261	0.1670	0.3794	0.2456	0.2982
SAFT_ISI Run 16	top-5 prelim. hop0+1 recall, diff. doc. only	0.2956	0.1388	0.1889	0.1951	0.0704	0.1034	0.2676	0.1159	0.1618

Table 1: Evaluation results for English ensembling, all scores are SF-All-Micro

Above we focused on the discussion of Hop-0+1 results for our submissions. Note, however, that Hop-0 results show significantly higher improvements. Our Run 1 improves by about 3.1 points over the best performing Team Run A. Moreover, Run 3 has similar f1 as the best-performing Team Run A but 13 points higher recall, which is significant, since from past evaluations it is clear that higher recall is generally more difficult to achieve. This is also about 9.5 points higher than the highest Hop-0 recall Run F. However, given the scoring strategy for Hop-0+1 results, it is clear that teams generally optimize towards higher precision for Hop-0 results to achieve better overall Hop-0+1 results, therefore, we are most probably not seeing the highest Hop-0 recall results possible.

Run 11 uses knowledge not available at evaluation time (shown in italics and grayed out), by selecting inputs based on their ranking from the final Hop-0+1 recall scores. Given this better ranking, we can improve the overall ensemble score by another 2.3 points to 31.09%, which shows that a better relative recall estimate of available inputs leads to better ensemble run performance. Runs 12 and 13 investigate how selecting based on f1 instead of recall affects the results, and both of them perform lower than their analogues selected based on recall. Run 14 takes all five submitted runs from top-scoring Team A and treats them as if they had come from different teams. We do not get any improvements, in fact results are below the average result of the five runs, showing that there is a lot less independence of results and errors within a team's submissions than across teams. Finally, Runs 15 and 16 investigate the impact of individual independent evidence aspects, such as two results coming from two different teams vs. from different documents. Run 15 uses the same inputs as Run 9 (the corrected Run 1) and only outputs results if they are supported by at least two different teams. The resulting ensemble outperforms our best Run 1 and 9 by another point. Run 16 again uses the same inputs but only outputs results that are supported by at least two different documents. This ensemble has a much worse f1 which is to be expected, since there will generally be fewer results supported by multiple documents, leading to significantly lower recall. However, precision is also significantly lower which suggests that we might get additional coreference errors from EDL results as well as our value-based matching across documents, thus, suggesting an important future area of improvement.

		Hop-0			Hop-1			Hop0+1		
SFV Run	**Run Characteristics**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
SAFT_ISI Run 1	top-3 prelim. hop0+1 recall	**0.2391**	0.0374	0.0647	0.0000	0.0000	0.0000	**0.2391**	0.0220	0.0402
SAFT_ISI Run 2	top-3 prelim. hop0+1 recall w/ EDL2	0.2391	0.0374	0.0647	0.0000	0.0000	0.0000	0.2391	0.0220	0.0402
CS SF/KB Run										
Team Run A	best final hop0+1 f1	0.1458	0.2653	**0.1882**	0.0000	0.0000	0.0000	0.1458	0.1557	**0.1506**
Team Run B	best final hop0+1 recall	0.0393	**0.3231**	0.0701	0.0000	0.0000	0.0000	0.0393	**0.1896**	0.0651
Team Run C	2nd best final hop0+1 f1	0.1765	0.0102	0.0193	0.0000	0.0000	0.0000	0.1765	0.0060	0.0116
Team Run D	3rd best final hop0+1 f1	0.2222	0.0068	0.0132	0.0000	0.0000	0.0000	0.2222	0.0040	0.0078

Table 2: Evaluation results for Spanish ensembling, all scores are SF-All-Micro

		Hop-0			Hop-1			Hop0+1		
SFV Run	**Run Characteristics**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
SAFT_ISI Run 1	top-3 prelim. hop0+1 recall	0.4561	0.0399	0.0733	0.3182	0.0341	0.0617	0.4177	0.0385	0.0705
SAFT_ISI Run 2	top-3 prelim. hop0+1 recall w/ EDL2	0.4561	0.0399	0.0733	0.3182	0.0341	0.0617	0.4177	0.0385	0.0705
CS SF/KB Run										
Team Run A	best final hop0+1 f1	0.5200	**0.1994**	**0.2882**	0.2905	**0.2537**	**0.2708**	0.4242	**0.2124**	**0.2830**
Team Run B	best final hop0+1 recall	0.5200	0.1994	0.2882	0.2905	0.2537	0.2708	0.4242	0.2124	0.2830
Team Run C	2nd best final hop0+1 f1	**0.6667**	0.1319	0.2202	0.3103	0.0439	0.0769	**0.6013**	0.1109	0.1872
Team Run D	3rd best final hop0+1 f1	0.4070	0.0537	0.0949	**0.3333**	0.0146	0.0280	0.4000	0.0443	0.0798

Table 3: Evaluation results for Chinese ensembling, all scores are SF-All-Micro

Table 2 summarizes scores from our two submitted Spanish SFV ensemble runs SAFT_ISI_1 and 2 and a number of best team runs A to D based on various scoring aspects shown in the Run Characteristics column. The problem with the Spanish language condition was that there were only submissions from three different teams available with extremely unbalanced performance. The top team run had an overall f1 of 0.15 while the best run from the second-best scoring team only had an overall f1 of 0.01. This causes a problem for our ensembling strategy which requires reasonably balanced recall among at least two of the best-scoring selected runs, and we therefore have a significant loss in recall in our top-scoring Run 1. Despite a significant boost in precision from 0.145 to 0.239, the loss in recall drops our f1 way below the performance of the best-scoring run. Our second ensemble Run 2 used a different EDL system which gave identical performance.

Table 3 summarizes scores from our two submitted Chinese SFV ensemble runs SAFT_ISI_1 and 2 and a number of best team runs A to D based on various scoring aspects shown in the Run Characteristics column. Our two ensembling runs (the second one again with a different EDL system) performed identically but much worse than the best-performing team runs. The Chinese language condition had much better team participation and balanced performance compared to Spanish which made it well-suited for our approach. However, our filler-token-based value clustering did not work at all for Chinese characters beyond strict identity which gave us very low recall. Unfortunately, we did not have time to do any development or testing specifically for Chinese before the evaluation which might have made us aware of this problem ahead of time. We expect the results to improve significantly with a filler clustering component adapted for Chinese.

4 Conclusion

We described a new slot filler ensembling component we developed as part of our Knowledge Resolver relation extraction and knowledge base population toolkit (KRes), and its performance on the TAC-KBP 2016 ensembling task. At the core of our approach is a process of *anti-error compounding* which identifies likely redundant extractions that are based on at least partially independent evidence, and aggregates them into more reliable results exploiting the redundancies. Our system does not require any training and only needs some approximate relative performance ranking of candidate input runs. By merging the top-5 recall KBs created by different teams based on preliminary score estimates distributed with the TAC-KBP 2016 SFV evaluation package, our system improved the best overall English SF-All-Micro f1 score for Hop-0 queries from 0.319 to 0.350, and for Hop-0+1 queries from 0.270

to 0.286. Our approach is language independent, however, for Spanish and Chinese we could not create any improvements due to a lack of run diversity for Spanish and a token-based name and value matcher that was not working sufficiently for Chinese.

One problem with the current formulation of the TAC-KBP ensembling task is that it is based on Cold Start knowledge bases which generally only report the best extraction of a particular slot filler as opposed to all of them above a certain confidence. This eliminates a significant amount of redundancy that would otherwise be exploitable by our approach. An interesting direction for future work is to see whether preserving such multiple extractions even if they are lower confidence could boost overall performance of an ensemble run.

Acknowledgment

This report is based on research sponsored by the Defense Advanced Research Projects Agency (DARPA), the Air Force Research Laboratory (AFRL) under agreement number FA8750-12-2-0342. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA, AFRL or the U.S. Government.

References

H. Chalupsky. 2012. Story-level inference and gap filling to improve machine reading. In *Proceedings of the Twenty-Fifth International FLAIRS Conference*. AAAI Press.

H. Chalupsky. 2013. English slot filling with the Knowledge Resolver system. In *Proceedings of the 2013 Text Analysis Conference (TAC 2013)*. NIST.

H. Chalupsky. 2014. English slot filling with the Knowledge Resolver system. In *Proceedings of the 2014 Text Analysis Conference (TAC 2014)*. NIST.

H. Chalupsky. 2015. Cold start knowledge base population with the Knowledge Resolver system for TAC-KBP 2015. In *Proceedings of the 2015 Text Analysis Conference (TAC 2015)*. NIST.