New York University 2012 System for KBP Slot Filling

Bonan Min Xiang Li Ralph Grishman
Computer Science Department
New York University

New York, NY 10003
{min xiangli grishman}@cs.nyu.edu

Ang Sun
Intelius
500 108th Ave NE Suite 2200
Bellevue, WA 98004
(work done at New York Univ.)
asun@intelius.com

Abstract

This paper describes the New York University 2012 system for the KBP regular slot filling (SF) task. The NYU 2012 SF system has a similar architecture to the NYU 2011 system. We improved our distant-supervision based slot-filling component with a few techniques including filtering errors by statistical measures collected from the source corpus, and relabeling erroneous training examples with a set of maximum entropy models and by applying bootstrapped/hand-coded patterns. We also augmented our query expansion procedure. After the formal evaluation we experimented with models for estimating slot confidence. We report on the impact of these changes.

1. Introduction

This paper describes the New York University 2012 system for the Knowledge Base Population (KBP) regular slot filling (SF) task, part of the Text Analysis Conference (TAC) organized by NIST. The NYU 2012 KBP Slot-filling system has a similar architecture to our 2011 system. The system consists of several slot-filling components: two that use hand-coded patterns, another pattern-based slot-filler in which the patterns are generated semi-automatically with a bootstrapping procedure, one based on name coreference, and a distant-supervision based slot filler. Significant improvements were made in the distant-supervision based slot-filling component using a diverse range of techniques. In particular, we used statistics gathered from the source corpus to measure the quality of relation argument pairs and to remove noise, and applied a set of maximum entropy models and bootstrapped lexical and dependency path patterns to refine the labels for "distantly" matched training examples.

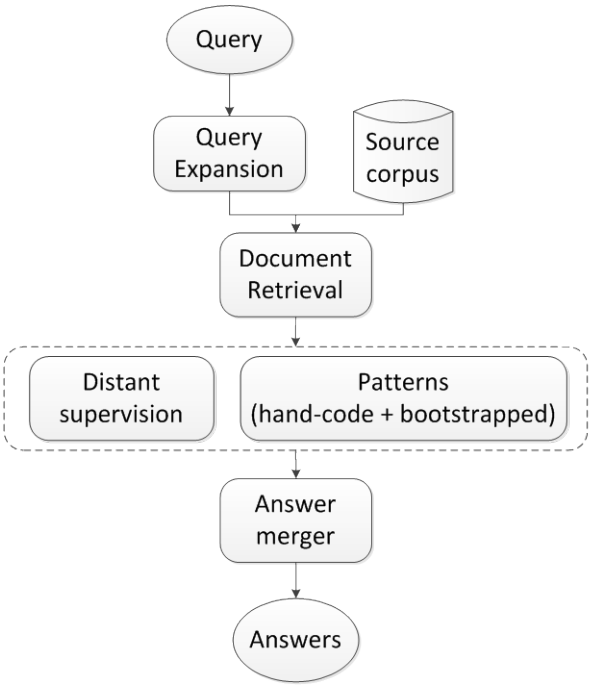
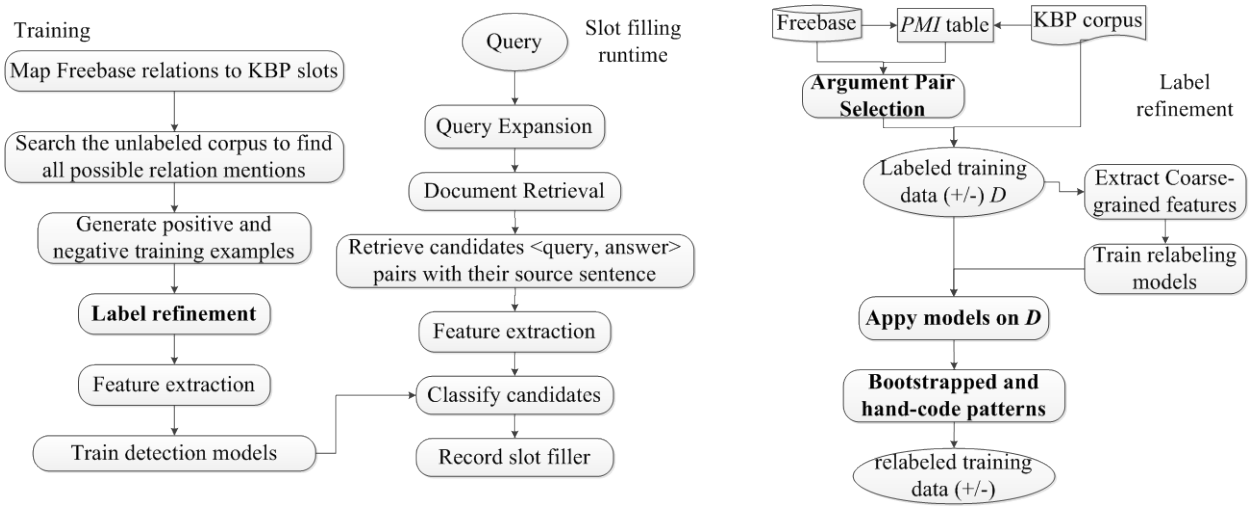


Figure 1. System overview



(a) Overview of distant-supervision component (b) zoom-in view of the label refinement procedure

Figure 2. Overview of NYU 2012 KBP slot filling component trained with distant supervision. a) shows the overview of distant-supervision based slot filling component, b) shows the label refinement sub-component, in particular the various techniques and how they interplay.

2. Overview of NYU Slot-Filling System

Like most KBP Slot-Filling systems, the NYU system starts by retrieving related documents based on a match to the query name or the result of query expansion. It then uses a set of five extractors (including one extractor trained with distant supervision) operating in parallel on the retrieved documents to extract fillers. The result is a set of *intermediate* slot fills, potentially highly redundant. Finally the system uses a combiner to validate answers and remove duplicates. Figure 1 shows a highly simplified architecture of the NYU KBP slot filling system. We refer the readers to our 2011 and 2010 system papers for more details (Sun et al. 2011, Grishman and Min 2010).

The major changes in the NYU system for 2012 involved improvements in distant supervision (described in section 3) and in query expansion (described in section 4). A small change was made to add a regular expression for Government titles, such as "Acting Deputy Assistant Secretary of State". A component for finding organization names in context was removed; it had contributed negatively to our 2011 performance, and we didn't forgive it for that transgression. An ablation study of the contribution of the remaining components is given in section 5.

Other minor changes were needed to track character offsets for fills and to report a confidence for each fill. For the formal runs, the confidence was simply the average precision on the 2011 evaluation of the extraction component which was responsible for the slot fill. Subsequent to the 2012 formal evaluation we began work on a more elaborate estimate of confidence; this is described in section 6.

3. Improved Distant Supervision

The largest changes in the NYU 2012 system over the 2011 system are in the distant-supervision-based slot-filling component. In this section, we first briefly review our distant-supervision-based slot filling component, and then describe the improvements in the NYU 2012 system. As usual, we refer the reader for any missing details of our slot-filling component to our 2011 system description paper (Sun et al, 2011).

3.1. Distant Supervision

The overall architecture is the same as the architecture of the NYU 2011 system, except that the label refinement procedure is replaced with a

new subsystem which applies a diverse range of techniques for the refinement. The overall architecture is shown in the Figure 2(a).

Training: Following the NYU 2011 system, the training procedure uses Freebase as the training source and the KBP source corpus as the unlabeled corpus for distant supervision. The same mapping table from Freebase to KBP slots is reused this year. An offline training procedure processes all documents in the source corpus, enumerates all entity pairs that appear in the same sentence, and extracts them with their reference sentences as candidate relation mentions. A separate offline procedure runs the Stanford parser[1] over the entire corpus to generate analyzed documents which contain part-of-speech tags, dependency parses, etc. These are used later on for extracting features.

The key step for distant supervision (Mintz et al. 2009, Surdeanu et al. 2011) is to automatically label its training data using the training source (Freebase). We label a relation mention as positive if its argument pair appears in the related tables from Freebase, and we label the relation mention as negative if its argument pair $< i, j >$ doesn't appear in Freebase but some $< i', j >$ or $< i, j' >$ appears in Freebase and $i' \neq i$ and $j' \neq j$.

After generating labeled examples, each example is represented with a diverse set of lexical, syntactic and semantic features (a detailed description and examples of features are in the NYU 2011 KBP system paper), and then a set of maximum entropy classifiers are trained and used as relation detection models. Because the distant labeling process generates an extremely unbalanced class distribution, we follow our last year's practice: training a multi-class maximum entropy model for each pair of entity types, and down sampling the *OTHER* class to the same size as the positive class.

Slot Filling runtime: During test time, the main NYU slot filling system reads in the queries, performs query expansion based on resources mined from Wikipedia redirect text, and then calls an IR engine (Lucene) to retrieve related documents, followed by deep linguistic analysis of the document using the NYU Jet system[2] (part-of-speech tagging, chunking, name tagging, coreference resolution, etc.). After these steps, a set of <*query, candidate*> pairs are passed to the distant filler along with the supporting sentences. Distant filler then performs feature extraction (using the same feature space as used at training time), and then classifies <*query, candidate*> pairs with the model trained offline. Candidates that are classified as correct are emitted as an answer and sent back to the main NYU system for post processing.

Problems with Distant Supervision: The heuristic labeling process of distant supervision generates noisy class labels which will hurt performance. This is particularly true when matching Freebase to a corpus consisting largely of news (Riedel et al. 2010 reported a 31% error rate when mapping Freebase to a New York Times corpus). There are two types of errors. First, a relation mention whose argument pair bears a certain relation (according to Freebase) doesn't necessarily express the relation in its local context. For example, there is no *org:founded_by* relation expressed between the argument pair *Bill Gates* and *Microsoft* in the sentence *Bill Gates has declared war on Microsoft's insecure software.* However, it is labeled as positive when consulting Freebase. Second, because Freebase is highly incomplete, the labeling process labels lots of relation mentions as negative when in fact there is contextual evidence that a relation exists between the pair of arguments. We will call the two types of errors *false positive matches* and *false negative matches* in the rest of the paper.

Label Refinement: Since the labeling process is known to generate *false positive matches* and *false negative matches*, the NYU 2011 system uses a few rules that correct the label of an example to the most frequent class its dependency path is associated with in the corpus (a more accurate and detailed description can be found in our system paper from last year). This year, we have improved it by applying a diverse range of techniques. In particular, we used statistics gathered from the source corpus to measure the corpus-dependent relatedness of KB entries (relation argument pairs) and to remove noise, applied a set of coarse-grained maximum entropy models, trained on the examples generated by distant supervision, to relabel the noisy training instances, and used

[1] http://nlp.stanford.edu/software/lex-parser.shtml
[2] http://cs.nyu.edu/grishman/jet/license.html

bootstrapped lexical/ dependency path patterns and hand coded patterns to refine the labels for "distantly" matched training examples.

In the following few subsections, we describe a few improvements to refine the training data for our 2012 KBP slot filling sub-system. Figure 2(b) shows the specific refinement procedures and their relations in the NYU 2012 slot filling system.

3.2. Pair Selection

Freebase tables are not perfect. For example, <*Linda, British*> appears as an argument pair for *per:origin*. A common English first name such as *Linda* is ambiguous in terms of referring to a specific entity and is prone to causing *false positive matches*. Moreover, there is a gap between the sources of supervision (e.g., the tables in Freebase) and the reference corpus. It is not clear whether a pair of entities that bears a specific relation expresses that relation when the entities co-occur in a sentence in the source corpus.

We hypothesize that if a pair of entities co-occur more frequently in the same sentence (and co-occur less frequently with other entities), they are more likely to be correlated, thus bearing a relation in this corpus. We define sentence-level Pointwise Mutual Information (PMI) as follows:

$$pmi_{sent} = log \frac{C_{e_1 e_2} \times N}{\sum_{i=1}^{m} C_{e_1 e_i} \times \sum_{j=1}^{m} C_{e_j e_2}}$$

in which *m* is the total number of entities, *N* is the total number of pairs of entities that appear in the same sentence, and $C_{e_1 e_2}$ is the count of co-occurrences (in the same sentence) of a pair of entities e_1 and e_2.

Feature name	Feature value	description
ETwDpath	E21-E_ORGANIZATION-poss-E_PERSON	Conjunction of 1) order of argument, 2) entity types of arguments and 3) the dependence path in between
ETwTpath	E21-E_ORGANIZATION-'s-E_ PERSON	Conjunction of 1) argument order, 2) entity types of arguments and 3) the lexical sequence in between

Table 1. Features used in the coarse grained models for example <*Bill Gates, Microsoft*> as in the sentence *Microsoft's Bill Gates*

Sentence-level PMI for each pair of entities in the training source (Freebase) is calculated based on the source corpus, and then is used to remove KB entries for which pmi_{sent} is less than a threshold. This gives a statistical measure of the extent a pair of entities in Freebase tables is related to each other. Despite its simplicity, we can see that this assigns <*Linda, British*> a score of -3.18, much less than the 13.3 assigned to another *per:origin* instance < *Guujaaw, Haida*>.

In practice this filter should be used conservatively since otherwise it removes too many entries from the training source. We use $pmi_{sent} = 1.0$ as the threshold and removed around 10% of the training examples. On the 2011 assessment queries, we observe improvements in both precision (from 20.1% to 22.1%) and recall (from12.6% to 13%), with an 0.9% (from 15.5% to 16.4%) overall improvement in end-to-end F1 score.

3.3. Coarse-grained models for refinement

The second addition to the label refinement pipeline is the use of a set of statistical models instead of a few rules for label refinement. The 2011 NYU distant-supervision-based component relabels training examples to the most frequent type (if not the same) that their dependency-path patterns express in the automatically generated training dataset. We generalize the idea by first training a set of coarse-grained maximum entropy models on the automatically generated training dataset, then using it to relabel these automatically-generated examples. The high-level intuition is that, by leveraging the data redundancy, the procedure appropriately weights features by their likelihood of appearing in relation instances for a particular relations, and in turn, these feature weights helps us to decide which instances are more likely to express the target relation. This procedure is able to correct both *false positives matches* and *false negatives matches*. The final distant-supervision-based relation detection model

is trained with this new training set. Two types of features are used in the coarse-grained models: the entity type pair in conjunction with the dependency path between the 2 arguments and the order of arguments, and the entity type pair in conjunction with the lexical path in between and the order of arguments. An example of the features is shown in table 1.

We implemented this approach for relabeling training examples and tested it with 2011 data. Since this training/testing process makes use of the same data set which has a class imbalance problem, we trained a multi-class coarse-grained model for each pair of argument entity types and downsampled the *OTHER* class to be the same size as the rest combined, following the main distant-learning training/testing scheme. The model-based relabeling technique significantly improves performance over the baseline model (the end-to-end performance of the distant-supervision-based slot filler alone on 2011 assessment data has increased from 15.5% F1 to 22.3% F1). Using this refinement scheme alone also outperforms the refinement rules used in our 2011 system (F1 increases from 20.7% to 22.3%). Table 2 shows the results when varying the threshold on the coarse-grained models for refinement. It shows that when the threshold is lowered, it correct more examples (by default the labels of other instances are kept unchanged), resulting in significant improvements in both precision and recall.

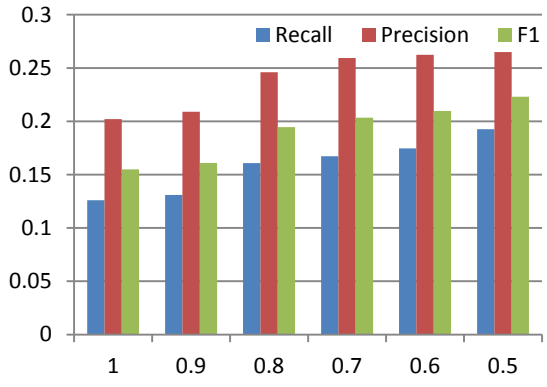


Table 2. Performance of distant-supervision based slot filling component with different threshold when applying the coarse-grained model for label refinement. "anydoc" is turned on in the scorer.

3.4. Patterns for refinement

The third change is that we seek ways to better combine the several slot-filling components instead of just merging the answers they emit. Specifically, we apply the bootstrapped and hand-coded patterns to correct the labels for "distantly" labeled examples that are used as the training source for the distant-supervision-based slot-filling component. This is based on the fact that our pattern-based slot-filling component has a higher precision than the distant-supervision based slot-filling component. Specifically, we correct the labels of the "distantly" generated examples that match a pattern generated by the bootstrapping procedure to the type associated with the pattern. The bootstrapped patterns correct around 30k labels while the hand-coded rules correct around 1 million labels. Overall, it improves overall F-measure around 4% compared to the baseline model (R/P/F1 improved from 12.6/20.2/15.5 to 17.2/25.6/20.6).

3.5. Other changes

Merge and re-divide similar slots. Observing that the 2 slots *per:employee_of* and *per:member_of* are very similar in their meanings, we treat them as the same class during the training and testing phase of distant learning, and we redivide them using a dictionary at the answer merging/validation phase.

To summarize the changes and see their relative improvements in the distant supervision component over our 2011 system, we did separate experiments with each change and listed them in Table 3.

	Precision	Recall	F1
Baseline (BL)	20.2	12.6	15.5
BL+model	26.5	19.3	22.3
BL+PMI filter	22.1	13	16.4
BL+patterns	25.6	17.2	20.6
BL+all	31.2	18.7	**23.4**
NYU 2011 DS	34.6	14.8	20.7

Table 3. The performance with each improvement over baseline distant supervision. **BL** is the naïve distant supervision with only undersampling on the *OTHER* class. **BL+model** uses the coarse-grained model for relabeling training examples. **BL+PMI filter** uses PMI to filter pairs and **BL+patterns** uses bootstrapped and hand-coded patterns for

refinement. **BL+all** uses all three techniques (illustrated in figure 2b). We also compare to our distant supervision based filler last year (**NYU 2011 DS**). "Anydoc" is turned on in the scorer.

Table 3 shows that the model-based refinement scheme improves the performance the most, followed by the pattern-based approach, and then the filtering approach based on PMI. All three approaches have a positive impact on the performance. The final combined distant-supervision-based slot-filling component has a F1 score of 23.4, outperforming the component in our 2011 system by 2.7 F1 score.

4. Query Expansion

Usually, it is not sufficient to retrieve all the documents related to the query by only using an exact match to the query's name string. Hence, we have implemented several ways to expand the query names, which can help improve the recall score of our Slot Filling system. Moreover, if these expanded query names have been validated in the corpus, they can be directly reported as the responses for the *per:alternate_names* slot or *org:alternate_names* slot. A query name can be expanded in the following ways:

1. If the query is an organization, the company suffixes can be dropped. For example, "*Patterson-UTI Drilling*" is considered as an alternate form of the query name "*Patterson-UTI Drilling Co.*";
2. If the query is a person with a middle initial, generate an alternate form by eliminating its middle initial(s);
3. A redirect dictionary similar to the one in Chen et al. (2010) has been generated from the Wikipedia database. Wikipedia uses the redirect links to indicate the target pages that refer to the same entity. The redirects for a query are used as the expanded query names, which may provide several name variations, nicknames, acronyms, full name, or alternate names for a query name. For instance, the redirects "*Parren James Mitchell*" and "*Parren J. Mitchell*" are expanded query names for the person query name "*Parren Mitchell*"; the expanded names for the organization query name "Asian Development Bank" are "AsDB" and "Asia Development Bank" from this redirect dictionary.

From the following table, we can observe that our strategies for query expansion have helped improve the recall score of our SF system by around 4%; the F1 score also increased by around 4%, mainly benefiting from the recall enhancement. Steps 1 and 2 were included in prior NYU systems; step 3 was added this year.

	Precision	Recall	F1
NYU1 w/o QE	42.6 (46.6)	18.0 (19.7)	25.3 (27.7)
NYU1 (step 1+2)	43.1 (46.5)	18.5 (20.0)	25.8 (27.9)
NYU1 (step 1+2+3)	**45.0 (47.0)**	**22.0 (23.0)**	**29.6 (30.9)**

Table 4. Effect of query expansion (QE). The numbers in parentheses are the scores with "anydoc" turned on.[3]

5. Results

NYU submitted 2 runs: one (NYU1) with the distant supervision models trained with the enlarged 2012 source corpus and refinement schemes and the other (NYU2) with the model trained last year. The official performance is shown in the first two rows of the following table. The overall F1 for the two runs is essentially equivalent. In terms of component level performance, the DS model we used this year outperforms the 2011 DS model.

	Precision	Recall	F1
NYU1	45	22	29.6
NYU2	48.9	21.2	29.6
NYU1 DS	33.3 (41.7)	6.9 (8.6)	11.5(14.4)
NYU2 DS	40.4 (50.8)	4.7(6)	8.5 (10.7)

Table 5. Official results of NYU runs. The NYU1 DS and NYU2 DS are the performance with the distant-supervision-based slot filler alone. The

[3] *anydoc* is a feature of the scorer which accepts a slot fill in the system response as correct, regardless of the document cited in support of the fill, so long as the assessment file (key) marks this fill as correct for some document.

numbers in parentheses are the scores with "*anydoc*" turned on.

To understand the impact of different components, we did an ablation study on the components, running/removing the components one at a time, and show the scores in Table 6. It shows that all of our components have a positive impact on the final performance, with local patterns and distant supervision being the highest performing components. Each of these two components alone has a performance exceeding the median score of all submissions.

Module	Scoring using only module			Scoring excluding module		
	Precision	Recall	F1	Precision	Recall	F1
Distant supervision	33.3	6.9	11.5	50	18.2	26.7
Alternate names	38.8	2.6	4.8	46.5	20.2	28.2
Local patterns	47.36	9.3	15.6	43.6	16.8	24.2
Bootstrapped linear patterns	59.2	4.6	8.5	43.8	20	28
Bootstrapped dependency patterns	54.8	3.7	6.9	43.1	19.9	27.2
Functional nouns	55.7	2.2	4.2	44.4	20.5	28

Table 6. Ablation study on NYU1

6. Confidence Estimation

6.1. Introduction

One fundamental problem for Information Extraction systems, including slot filling, is evaluating the probability that the extracted information is correct. Many current KBP SF systems, including our own, consist of several independent extraction pipelines. The system combines the responses from each pipeline; if a confidence value can be associated with each response, it can help re-rank/combine the responses. For this purpose we require comparable confidence values from disparate machine learning models or different slot filling strategies. We have employed a simple but effective confidence estimation model to solve this challenge. This model can incorporate the local features, pipeline features and global features to approximate the confidence value for each response under a consistent and uniform standard.

6.2. Framework

This confidence estimation model mainly applies the following three general categories of features. Then it trains a Maximum Entropy model using these features and assigns the probability of the response being correct as the confidence value for that response. Three general categories of features are used:

(1) Local Features: checks whether the supporting sentence contains both Query and Answer and also generates the dependency parse path related features;

(2) Pipeline Features: indicates how well each pipeline, which provides the response, performed previously;

(3) Global Features: detects how closely the Query and Answer are correlated in the global context.

Each specific feature in the above categories is listed in Table 1.

Feature Category	**Feature**	**Description**
Local Features	support_sent_contain_Q_A	Checks whether S contains both original Q and A;
	support_sent_contain_ExQ_A	Checks whether S contains both co-referred Q or expanded Q and A;
	shortest_dpath_length_Q_A	The length of shortest dependency parse path

		between *Q* and *A* in *S*;
	Shortest_dpath_Q_A	The shortest dependency parse path between *Q* and *A* in *S*;
Pipeline Features	slot_name	The slot name;
	pipeline_name	The name of pipeline which generates *A*;
	pipeline_precision	The precision of the pipeline which generates *A*;
Global Features	query_retrieve_doc_num	The number of documents retrieved by *Q*;
	answer_retrieve_doc_num	The number of documents retrieved by *A*;
	co-occurrence_doc_num	The number of documents retrieved by the co-occurrences of *Q* and *A*;
	cond_prob_answer_givenQ	The conditional probability of *A* given *Q*;
	cond_prob_query_givenA	The conditional probability of *Q* given *A*;
	point-wise_mutual_info	The Point-wise Mutual Information (PMI) of *Q* and *A*;

Table 1. Features of Confidence Estimation Model.
(Notation: *Q* -- query; *A* -- answer candidate; *S* -- supporting sentence.)

6.3. Experiment

To evaluate the reliability of confidence values generated by this model, we used the weighted voting method to investigate the relationship between the confidence values and the performance in Precision(P) / Recall(R) / F-measure(F) scores. We trained this confidence estimation model using one year's SF evaluation data, and then applied the model to estimate the confidence values for all the intermediate responses generated for another year's data.

Baseline voting system: Both NYU1 and NYU2 SF systems apply a basic voting system to combine all intermediate responses to generate the final response submission. This voting system simply counts the number of each response entity, which is a unique response tuple in the form <Query_ID, Slot_Name, Response_Fill>, reported by all pipelines. For a single-valued slot of a query, the response with the highest count is returned as the final response fill. For the list-valued slots, all the intermediate responses extracted by the pipelines are returned as the final response fills. In this basic voting system, each response contributes equally.

Weighted voting system: Weighted voting is a voting system based on the idea that not all the voters contribute equally. Instead, voters have different weights concerning the outcome of an election. In our experiment, voters are all of responses generated by all pipelines, and the voters' weights are the confidence values of those responses. We also set a threshold *t* in this weighted voting system, where those intermediate responses with confidence values that are lower than *t* would be eliminated. For each response entity, this weighted voting system simply sums all the weights (confidences) of the intermediate responses that support this response entity as the weights of this response entity. Then for a single-valued slot of a query, it returns the response with the highest weights as the final response fill, while it returns all the responses as the final response fills for the list-valued slots.

6.4. Results

We first trained a confidence model on KBP2010 SF evaluation data and generated confidence values for each intermediate response on KBP2011 SF evaluation data. The above weighted voting system is applied to combine these intermediate responses with their confidence values. When the threshold t was set to 0, we found the scores of this weighted voting system result are 0.310/0.317/0.313 in P/R/F, compared to the baseline voting system result 0.296/0.306/0.301. If we varied the threshold t, the best result was 0.397/0.288/0.334 in P/R/F, which improved the baseline result by around 3% (absolute) in F-measure.

We then trained the confidence model on the 2011 data and applied it to the 2012 data. When

we set confidence threshold `t` to 0, the P/R/F scores of the final responses produced by this weighted voting system are 0.473/0.231/0.311, compared to the P/R/F scores, 0.470/0.230/0.309, of NYU1 on KBP2012 SF data (where "anydoc" was turned on for both cases). Raising the threshold did not yield any further improvement. The reasons why this weighted voting system performs much better on KBP2011 SF evaluation data than KBP2012 SF evaluation data still need to be investigated.

Figure 3 summarizes the results of this weighted voting system with different confidence threshold settings. When the confidence threshold is raised, the precision score continuously increases to 1, and the recall score gradually decreases to 0.

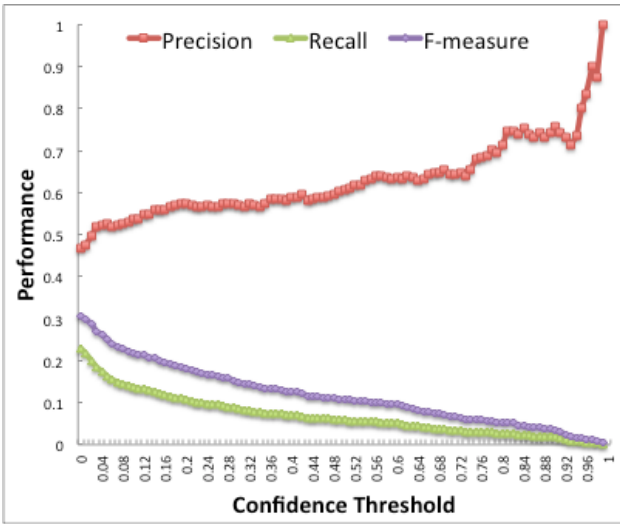


Figure 3. Impact of Confidence Threshold Settings

In addition to improving overall performance, the confidence estimates can be used to convey to the user of slot-filling output our confidence in individual fills. We divided the range of confidence values (0 to 1) into 10 equal intervals (0 to 0.1, 0.1 to 0.2, etc.), and then categorized the intermediate responses by their confidence value. Then for each category, the intermediate responses are combined by the above weighted voting system, and the final response fills are scored in Precision score. Figure 4 shows the main tendency for the responses with higher confidence to generate more precise answers, indirectly validating the reliability of the confidence values.

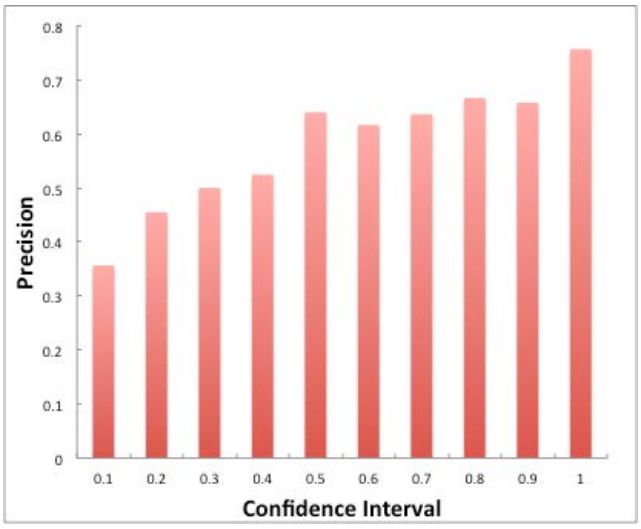


Figure 4. Performance of responses in each confidence interval.

7. Conclusion

This paper describes the NYU 2012 system for the KBP slot filling task. We have improved our distant-supervision-based slot-filling component with several novel techniques. Results show improvement over our distant-supervision-based slot-filling sub-system last year. We augmented query expansion, which substantially improved this year's performance. We analyzed the NYU runs this year, and presented an ablation study to understand the impact of each component and their relative contribution to the final system. Finally, we created a preliminary model for estimating our confidence in slot fills and showed its correlation with precision; further experiments are planned to investigate more elaborate models.

Acknowledgments

Supported in part by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20154. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

References

Zheng Chen, Suzanne Tamang, Adam Lee, Xiang Li, Wen-Pin Lin, Javier Artiles, Matthew Snover, Marissa Passantino and Heng Ji. 2010. CUNY- BLENDER TAC-KBP 2010 Entity Linking and Slot Filling System Description. *Proceedings of Text Analysis Conference 2010*.

Ralph Grishman and Bonan Min. 2010. New York University KBP 2010 Slot Filling System. *Proceedings of Text Analysis Conference 2010.*

Mihai Surdeanu, Sonal Gupta, John Bauer, David McClosky, Angel X. Chang, Valentin I. Spitkovsky, and Christopher D. Manning. 2011. Stanford's distantly supervised slot-filling system. *Proceedings of the Text Analytics Conference 2011*.

Mike Mintz, Steven Bills, Rion Snow and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. *Proceedings of ACL-IJCNLP 2009.*

Sebastian Riedel, Limin Yao and Andrew McCallum. 2010. Modeling Relations and Their Mentions without Labeled Text. *Proceedings of ECML/PKDD 2010*.

Ang Sun, Ralph Grishman, Wei Xu and Bonan Min. 2011. New York University 2011 System for KBP Slot Filling. In *Proceedings of Text Analysis Conference 2011*.