New York University 2014 Knowledge Base Population Systems

Thien Huu Nguyen, Yifan He, Maria Pershina, Xiang Li, Ralph Grishman
Computer Science Department
New York University
{thien, yhe, pershina, xiangli, grishman}@cs.nyu.edu

Abstract

New York University (NYU) participated in three tracks of the 2014 TAC-KBP evaluation: English Slot Filling, Cold Start and Entity Discovery and Linking. While this year is the first time and second time we participated in entity discovery and linking (EDL) and cold start respectively, we have been working on the slot filling task for several years. With additional development time this year, our cold start system has borrowed more technologies from slot filling and gained significant improvement. In both slot filling and cold start systems, we find that besides incorporating new modules to expand the coverage, the systems greatly benefit from our effort to renovate some of the basic modules. Regarding entity discovery and linking, we develop a pipelined system with several components in which we introduce a variation of the PageRank algorithm to improve collaborative candidate ranking.

1 Slot Filling and Cold Start: Overview

The NYU KBP slot filling and cold start systems for 2014 inherit the structure of our systems from the last several years (Sun et al., 2011; Min et al., 2012; Grishman, 2013). We improved several of our basic components: the name tagger, the relation extraction components based on hand-coded patterns, and distant supervision. These are the common components for both the slot filling and cold start systems. Note that the distant supervision module is new to the cold start system; because of time constraints, we only used pattern-based relation extraction modules in the system last year.

We introduce new inference modules to both cold start and slot filling systems this year. Although the inference components differ, they share the same background of manual rules.

In addition, for slot filling only, we added one more distant supervision module employing the new guided multi-instance multi-label distant supervision framework we have developed recently (Pershina et al., 2014). Finally, a named entity disambiguation module is adopted to handle ambiguous names. An overview of our slot filling and cold start systems can be found in figures 1 and 2.

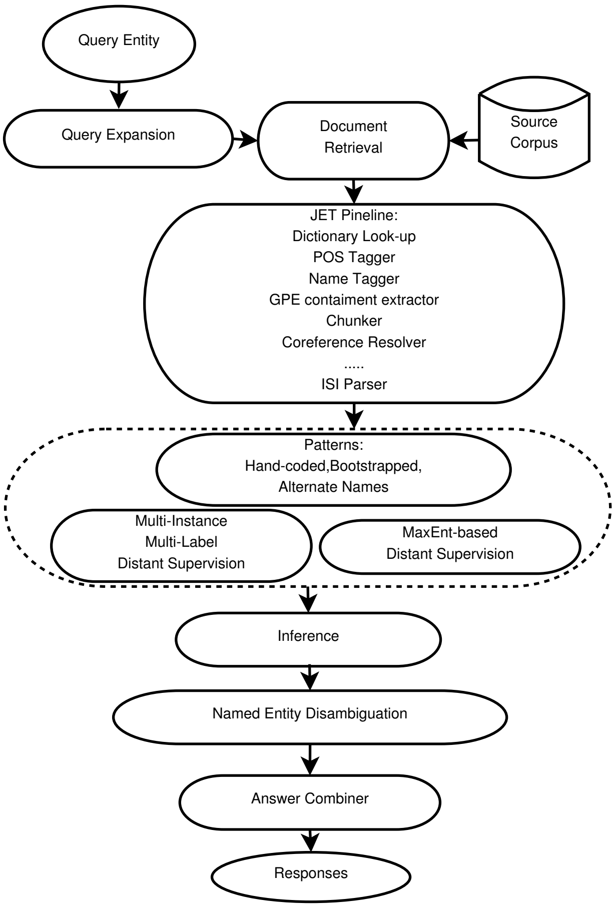

Figure 1: The NYU 2014 system for slot filling.

In the following, we describe the changes we made to the basic components (also the common components) as well as the architecture of the new

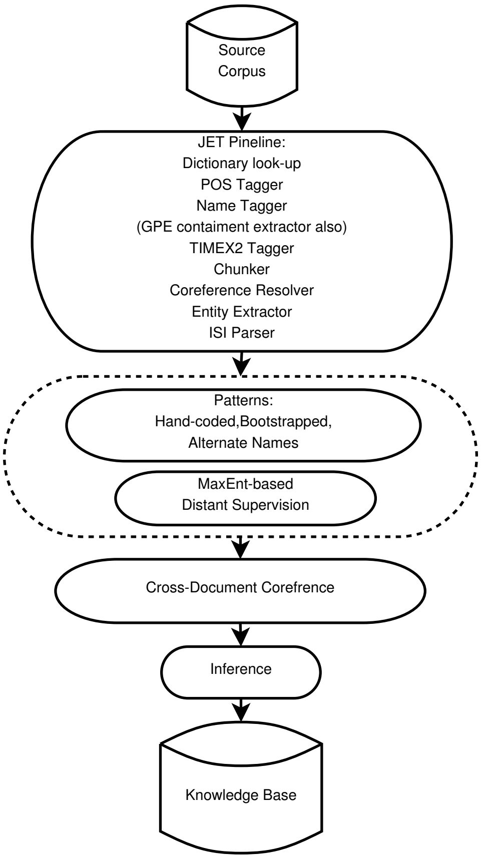

Figure 2: The NYU 2014 system for cold start.

modules. Finally, the system performance is reported.

2 Common Components for Slot Filling and Cold Start

2.1 Name Tagger

For prior years, our name tagger was trained on the ACE dataset without using word clusters as features. This year, we modified the OntoNotes dataset to be compatible with ACE data (with respect to named entities) and re-trained the name tagger on the two datasets. We also build large-scale word clusters on the TAC corpus as well as the ACE and OntoNotes dataset and utilize these word clusters as additional features for the name tagger. Although we do not report how much the new name tagger helps the system independently in the experimental result section, our experience from the evaluation on the 2013 data is that the new name tagger has an impact on every relation extraction module and contributes roughly 4 percent absolute improvement to the final system.

2.2 Hand-coded Patterns

In the slot filling and cold start components based on hand-coded patterns of previous years, we used a 1900-pattern set of word sequences and dependency paths. This year, we expand this pattern set by applying the paraphrase database (PPDB) (Ganitkevitch et al., 2013) which was extracted from bilingual parallel corpora. In particular, we first search the database for paraphrase pairs that contain some phrase in our original pattern set. After that, the other phrases of the matched pairs are returned for manual review and added to the original pattern set. Moreover, we introduce some new patterns discovered from our examination of additional documents which eventually produces a set of 3000 patterns. This new pattern set is also applied in our cold start system.

2.3 MaxEnt-based Distant Supervision Module

For our prior slot filling systems, the feature set of our distant supervision module included the dependency paths from the Stanford parser, which was quite slow[1]. This year we switched to the much faster Tratz-Hovy dependency parser from ISI (Tratz and Hovy, 2011) which makes it very convenient for us to integrate the MaxEnt-based distant supervision module into the cold start system, as we often need to parse the whole TAC corpus for features when we do the cold start evaluation.

For training data, we first align the relation tuples obtained from Freebase with text from the official 2010 KBP document collection[2]. The resulting relation instances are then refined by some heuristics (Sun et al., 2011) as well as the hand-coded patterns presented above to mitigate the impact of false positive and negative examples. In particular, for hand-coded patterns, we correct the labels of the “distantly” generated relation instances matching some pattern to the type associated with that pattern (Min et al., 2012). Finally, we use the MaxEnt framework to train relation extraction models for each pair of entity types (Sun et al., 2011) to be used in both slot filling and cold start.

[1] The Stanford parser recently added a faster, transition-based dependency parsing module, which we plan to experiment with in the future

[2] This combination gives us the best performance during the evaluation on the 2013 data.

Number of training instances generated by alignment	130570
Number of negative training instances generated by alignment	51656
Number of positive training instances generated by alignment	78914
Number of entities in training data (groups of instances)	78266
Number of entities with more than one filling mentions	19541 (24.97%)
Number of positive entities (positive groups)	30913
Number of negative entities (negative groups)	47353
Number of labeled sentences in the 2012 evaluation data	1705
Number of distilled guidance patterns	926

Table 1: Training data and guidance pattern statistics for MIML.

3 Rule-based Inference Modules for Slot Filling and Cold Start

For slot filling, this module basically implements geo-political containment rules to automatically fill in the slots of this type. For instance, once one of the slot extraction modules can extract "*Cupertino*" as the filler of the slot "*org:city_of_headquarters*" for the organization entity "*Apple*", and once we can collect evidence somewhere in the documents saying that "*Cupertino*" is a city of the U.S state "*California*", we would then report "*California*" as the filler for the slot "*org:stateorprovince_of_headquarters*" of "*Apple*". This reasoning step is performed after the system has finished all the other slot filling modules for each query entity. In order to gather the evidence of geo-political containment, we designed some high-precision rules and apply these rules when we extract names from the retrieved documents for each query entity. This evidence is accumulated during the course of the system run so that the evidence recognized during the processing of some query entity can be used later for the next entities. We utilize a small gazetteer to decide whether a geo-political entity is a city, a state, a country or other type of entity.

For cold start, we employ a larger set of forward chaining inference rules, in which, besides geo-political containment, the rules are mainly based on family relationships. The containment relations are acquired in the same way as for slot filling but on the whole evaluation corpus. Again, the inference module is a post-processing step and takes all the relations discovered by the relation extraction modules as inputs.

4 New Modules for Slot Filling

This section describes the new modules we build only for the slot filling system this year. Due to the limitation of time, we have not been able to integrate these modules into this year's cold start system.

4.1 Guided Multi-Instance Multi-Label Distant Supervision Module (MIML DS)

We integrate a new distant supervision module grounded on the guided multi-instance multi-label setting into the slot filling system. The core of this component is the multi-instance multi-label learning framework with the EM algorithm described in (Surdeanu et al., 2012). We generate training data for this framework via the alignment of relation tuples adapted from attributes of entities of the 2009 KBP evaluation reference knowledge base (essentially Wikipedia info boxes) with the snippets of wikipedia articles extracted from this knowledge base itself[3]. Most of the pre-processing tasks (name tagging, part-of-speech tagging, reference resolution etc) on the data are performed by our JET toolkit[4] except that we again parse the text by the dependency parser from ISI (Tratz and Hovy, 2011).

During the course of the EM algorithm, in the E steps, we re-label some relation instances following the guidance mechanism presented in (Pershina et al., 2014). The guidance patterns are essentially the de-lexicalized dependency paths (optionally accompanied by an important context word) and distilled from the 2012 KBP slot filling evaluation data. The rationale for the de-

[3] This alignment is due to the limited time we spend on this module. We believe that the model would be improved further if we obtain more training data from the alignment with other corpus such as the TAC corpus, etc.

[4] `http://cs.nyu.edu/grishman/jet/jet.html`

lexicalized dependency paths is our expectation for more general patterns obtained from such a small labeled dataset of the 2012 evaluation data. In our experience, generating more guidance patterns by exploiting a larger labeled dataset or active learning methods would be highly effective in improving the model performance but due to time limitations was not possible for the formal evaluations this year. Some statistics for the training data and guidance patterns we used for the system are described in Table 1.

Our feature set includes the following features: the context words between and around the two entity mentions, the entity types of the two mentions, the order of the two mentions, and the shortest de-lexicalized dependency path connecting the two mentions. This feature set is smaller than the one in (Surdeanu et al., 2012) as we want to avoid overfitting over our smaller training dataset. In addition, for greater generality, we incorporate the word clusters of the words along the shortest dependency paths as additional features. These word cluster features are shown to be effective in our evaluation on the 2013 data.

4.2 Named Entity Disambiguation Module

As the entities for 2014 KBP slot filling are intended to be ambiguous, we apply the AidaLight algorithm (Nguyen et al., 2014) to perform named entity disambiguation as a post-processing step. This module takes the fillers of all the slot-filling and inference modules as the inputs and remove the fillers whose hosting entities are different from the evaluation entities.

5 Experimental Results for Slot Filling

5.1 Evaluation on the 2013 Data

We test our system on the 2013 KBP slot filling evaluation data. Table 2 presents the experimental results when we incrementally add the modules into the system while Table 3[5] reports the results of the ablation study of the system. For these experiments, we consider the hand-coded patterns, the patterns generated by bootstrapping and the alternate name module as belonging to a single pattern-based module (denoted by "patterns" in tables). All the scores in Table 2 and 3 are computed with the "*anydoc*" flag. We do not use the named entity disambiguation module here although the inference module is sill included.

As we can see from the tables, the system achieves the best performance of 40.37% when all the modules are used together, in which the pattern-based module contributes 36.68%, the MaxEnt-based DS and MIML DS modules then add 3.15% and 0.54% respectively. The module with the best performance is the pattern-based module; the new name tagger and patterns improve the system performance by 10.78% (comparing row 1 of Table 2 that shows the performance of the our pattern based module last year when the old name tagger was used and row 2 of Table 2 that shows the performance of the same module on the same data when the new name tagger and the new patterns are integrated). Regarding MIML DS modules, the guidance mechanism, unfortunately, does not help the overall system. This could be because the information introduced by the guidance patterns has been covered by either our pattern sets or the MaxEnt-based distant supervision module already. However, the guidance is actually useful for the MIML DS framework itself (the performance is improved from 23.44% to 26.29%).

5.2 Evaluation on the 2014 Data

For the 2014 evaluation data, our experimental scenario is the same as the 2013 data and the results are given in Tables 4, 5 and 6 where scores with and without the "*anydoc*" flag are indicated. We submitted 4 runs for slot filling this year whose ids are shown in Table 4.

In general, our qualitative comments regarding the system on the 2013 data are still true for the 2014 data except that the guidance mechanism no longer enhances the MIML DS framework in this case. The pattern based module itself achieves the F score of 22.66% while the MaxEnt-based distant supervision and MIML DS modules, in the presence of the inference module, add 5.30% and 1.10% respectively . For the inference module, although it does not ameliorate the pattern based module, it actually helps the MaxEnt-based distant supervision and MIML DS modules to improve the overall system performance (the margin is rather small, though). Finally, the current

[5] In Table 3, for modules based on patterns and MaxEnt-based distant supervision, in the columns for scores excluding modules, the scores in the left correspond to the setting where the MIML DS module (without guidance) is applied while the scores in the right correspond to the setting where the guided MIML DS module is employed.

Module		P	R	F
1	Patterns of the 2013 system	54.49	16.98	25.90
2	Patterns of the 2014 system	54.07	27.76	36.68
3	2 + MaxEnt-based distant supervision	51.13	32.63	39.83
4	3 + MIML DS without guidance	45.52	36.26	40.37
5	4 + Guided MIML DS	44.33	36.94	40.30

Table 2: The slot filling system performance when modules are added incrementally on the 2013 data.

Module	score using only module			score excluding module		
	P	R	F	P	R	F
Patterns	54.07	27.76	36.68	46.46/44.19	22.07/22.69	29.93/29.98
MaxEnt-based distant supervision	54.16	13.85	22.05	46.24/45.26	32.83/35.02	38.40/39.49
MIML DS without guidance	46.91	15.63	23.44	51.13	32.63	39.83
Guided MIML DS	44.94	18.57	26.29	51.13	32.63	39.83

Table 3: The ablation study of the slot filling system on the 2013 data.

Module		score WITHOUT anydoc			score WITH anydoc		
		P	R	F	P	R	F
1	Patterns (NYU4)	36.59	16.42	22.66	41.91	18.00	25.18
2	1 - Inference	37.39	16.52	22.91	41.67	17.62	24.77
3	1 + MaxEnt-based distant supervision	39.67	21.59	27.96	41.68	21.71	28.55
4	3 - Inference	39.56	21.53	27.88	41.70	21.52	28.39
5	3 + MIML DS without guidance (NYU3)	33.82	25.47	29.06	36.99	26.67	30.99
6	5 - Inference	33.42	24.85	28.50	36.49	25.71	30.17
7	3 + Guided MIML DS (NYU1)	31.07	25.07	27.75	34.28	26.48	29.88
8	7 - Inference	30.94	24.75	27.50	34.17	25.90	29.47
9	7 + Named entity disambiguation (NYU2)	32.16	24.58	27.86	35.42	25.90	29.92

Table 4: The slot filling system performance when modules are added incrementally on the 2014 data.

Module	score using only module			score excluding module		
	P	R	F	P	R	F
Patterns	36.59	16.42	22.66	27.11/23.29	11.84/11.54	16.48/15.44
MaxEnt-based distant supervision	38.22	7.26	12.21	30.72/27.73	21.09/21.00	25.01/23.90
MIML DS without guidance	17.86	5.97	8.95	39.67	21.59	27.96
Guided MIML DS	15.18	6.27	8.87	39.67	21.59	27.96

Table 5: The ablation study of the slot filling system on the 2014 data WITHOUT using anydoc.

Module	score using only module			score excluding module		
	P	R	F	P	R	F
Patterns	41.91	18.00	25.18	38.72/34.14	16.19/16.19	22.83/21.96
MaxEnt-based distant supervision	47.64	8.67	14.67	36.38/33.51	23.90/24.29	28.85/28.16
MIML DS without guidance	36.01	11.52	17.46	41.68	21.71	28.55
Guided MIML DS	31.57	12.48	17.88	41.68	21.71	28.55

Table 6: The ablation study of the slot filling system on the 2014 data using anydoc.

named entity disambiguation module (row 9 in Table 4) has moderate impact on the system with the guided MIML DS (row 7 in Table 4) (an improvement of 0.11%).

Last but not least, we observe a huge drop in the slot filling score from 2013 evaluation set to 2014 evaluation set. In order to understand this loss, we gathered statistics on the 10 entity queries designated "confusable" (used for slot filling and entity linking); these are entities: 14, 19, 22, 27, 31, 62, 76, 79, 96, 99. For these entities, on the system with the MIML DS module without guidance (row 5 in Table 4), we get a precision of 22.47% (with "*anydoc*" flag). Compared to the overall precision of 36.99%, we see that these confusable entities contribute to the performance loss significantly. This suggests that a named entity disambiguation module should be included into the system to make it more robust to the ambiguous entities.

6 Experimental Results for Cold Start

Table 7 presents the scores for both our current system and the last year system on the 2013 cold start evaluation data. The results in row 2 of this table do not include the contribution of the inference module to be reported for the 2014 data only. It is very clear from this table that our system have been improved significantly this year (by 6.79% on the 1-hop F score). This confirms the effectiveness of the newly integrated modules for our CS system (the new name tagger, patterns and the distant supervision modules).

NYU submitted two runs for the KB-variant CS evaluation this year whose results are shown in Table 8. NYU2 is the system consisting of all the newly introduced modules except the inference module and similar to the best system in Table 7 (the "current" system). As we can see, the performance of this system (NYU2) on the 2014 data is less than that on the 2013 data (though by only a small margin on the 1-hop scores), suggesting that the evaluation data this year is somehow more challenging that the prior year. In addition, when the inference module is added to NYU2, resulting in NYU1, we attain a performance of 10.33% that is 0.66% better than NYU2's and demonstrates the benefit of this module.

7 Entity Discovery and Linking: Models and Algorithms

Given a knowledge base (KB) and a set of documents, the Entity Discovery and Linking (EDL) task involves identifying entity mentions in the documents and linking them to corresponding nodes in the KB. For mentions that are not linked to any existing node in the KB, the linker should cluster mentions that refer to the same entity together. In the context of TAC, the KB is a 2008 dump of Wikipedia (official KB hereafter), and the source documents are a set of newswire/web/forum documents provided by task organizers.

We first index the recent Wikipedia dump and generate possible aliases for each Wikipedia title in the preprocessing phase (Section 7.1). Entity linking is performed on this enriched KB and is mapped back to the official KB later.

We illustrate the overall architecture of our EDL system in Figure 3. We generate queries from source documents with a named entity tagger plus heuristic rules (Section 7.2). After queries are generated (EDL) or provided (EL), we retrieve linking candidates through exact alias matching

	0-hop			1-hop			Combined		
System	P	R	F	P	R	F	P	R	F
Last year	45.81	7.28	12.56	16.35	1.96	3.50	35.15	4.99	8.74
Current	67.40	21.96	33.13	59.80	5.63	10.29	66.04	14.94	24.37

Table 7: The cold start system performance on the 2013 evaluation data.

	0-hop			1-hop			Combined		
System	P	R	F	P	R	F	P	R	F
NYU1	69.45	17.80	28.34	53.42	5.72	10.33	63.90	11.04	18.82
NYU2	70.93	17.05	27.50	52.14	5.35	9.71	64.32	10.50	18.06

Table 8: The cold start system performance on the 2014 evaluation data.

(Section 7.3) and pick the best candidate with the PageRank algorithm (Section 7.4). Finally, we perform one-name-per-cluster NIL clustering (Section 7.5).

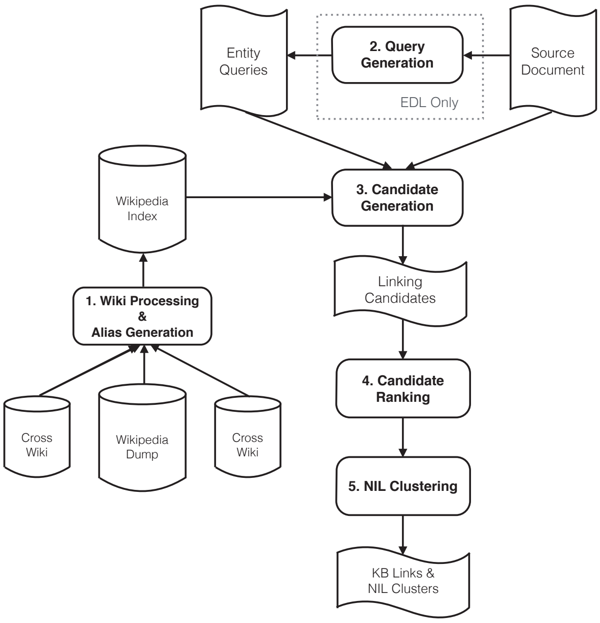

Figure 3: System architecutre of the NYU EDL system.

7.1 Wikipedia preprocessing and alias generation

We analyze a June 14, 2014 dump of Wikipedia and generate possible aliases for each entity in Wikipedia using Wikipedia redirects, Wikipedia disambiguation pages, CrossWiki (Spitkovsky and Chang, 2012), and WikiLinks (Singh et al., 2012).

Our alias generation process roughly follows that of Radford et al. (2012), except that we perform bootstrapping to remove much of the boilerplate text in CrossWiki, such as "Wikipedia article of . . . " and ". . . in Wikipedia".

We assign an integer id to each entry in the dump and store the id and name of each page along with its aliases, the ids of its linked entries, and the full text of the article in a Lucene index. Wikipedia entries are mapped back to the Official TAC KB using redirects and disambiguation pages.

7.2 Query generation

We use the same name tagger used in our SF and CS (cf. Section 2.1) submissions to identify PER/ORG/GPE names in the source document. Poster names in metadata or XML tags are identified with regular expressions. We then generate an EDL query for each name we identify.

To generate queries for nested names, we use several gazetteer-based rules on the output of the name tagger: 1) if an ORG name starts with a GPE name, we extract the GPE name; 2) if an ORG name ends with "at GPE", we extract the GPE name; 3) if an ORG name ends with "of GPE", we extract the GPE name; and 4) for the pattern "GPE1, GPE2", if GPE2 is the name of a country or US state, we generate a query for "GPE1, GPE2".

We do not have rules of other types of nested names, so the system is not able to generate a query for "Hertz" in "Hertz Foundation".

7.3 Candidate generation

We first expand names in the query and then search the Lucene index to generate candidates. Given a source document and a set of EDL/EL queries from this document, we perform named entity tagging and coreference resolution using the MEMM name tagger and the coreference resolution component in Jet[6]. If a name is not tagged by the tagger, we will add the name to the output of the name tagger. The name in the query is then expanded to its longest coreferent in the coreference chain.

If a name consists of only capitalized letters and dots and is not resolved to any other entity in the source document, we consider it to be a possible unresolved abbreviation. We then perform the abbreviation expansion algorithm of Schwartz and Hearst (2002) to determine if it is an abbreviation of another name in the article. We expand any abbreviation we find to its unabbreviated form.

We search the Lucene index of the Wikipedia dump to retrieve candidate entries for linking. We only retrieve candidates whose name or alias exactly matches the expanded name in the EL query. Matching is performed on normalized strings: letters are lowercased; accents and punctuation marks are removed. The maximum number of retrieved entries is set to 100.

7.4 Candidate ranking

The NYU system follows the PageRank approach proposed by Alhelbawy and Gaizauskas (2014) to rank candidates. Given a set of EL queries and a list of candidates for each query, we construct an undirected graph in which vertices are query-candidate pairs and an edge exists between two

[6] http://cs.nyu.edu/grishman/jet/jet.html

	Precision	Recall	F1
Flat tagger	0.694	0.687	0.691
Nested tagger	0.683	0.710	0.696

Table 9: Name tagging performance: `strong_typed_mention_match` on TAC14train

	Precision	Recall	F1
Flat tagger	0.711	0.690	0.700
Nested tagger	0.702	0.705	0.703

Table 10: Name tagging performance: `strong_typed_mention_match` TAC14eval

vertices if there is a hyperlink in either direction between the two candidate Wikipedia entries. All edges are assigned a uniform weight. We then run the PageRank algorithm on this graph and pick the candidate with the highest probability for each query.

We try to better utilize the initial score in PageRank algorithm and use the improved algorithm in some of our runs. We will report more details in a forthcoming paper.

7.5 NIL clustering

We first cluster together the queries that are linked to the same Wikipedia entry and then perform one-name-per-cluster NIL clustering.

8 Analysis of results

8.1 Experimental Settings

We used TAC EL 2012 (LDC2012E102, TAC12 hereafter) and 2013 evaluation data (LDC2013E90, TAC13 hereafter) for development and ran additional experiments on TAC 2014 EDL training data (LDC2014E15, TAC14train hereafter). The official evaluation scores were obtained by running our system on TAC 2014 EDL evaluation data (LDC2014E81, TAC14eval hereafter). We used a June 14, 2014 Wikipedia dump and linked it to the original TAC KB using redirects and disambiguation pages.

	Wiki F1	B$^3+$F1
Original PR: TAC12	0.743	0.710
Improved PR: TAC12	0.758	0.726
Original PR: TAC13	0.769	0.648
Improved PR: TAC13	0.779	0.664
Original PR: TAC14train	0.805	0.753
Improved PR: TAC14train	0.810	0.760

Table 11: EL results on training and validation sets

	Wiki F1	B$^3+$F1
Original PR (NYU3)	0.799	0.751
Improved PR (NYU1)	0.813	0.764
Indegree (NYU2)	0.814	0.766
Cosine (NYU4)	0.813	0.764
Lucene (NYU5)	0.815	0.765

Table 12: Diagnostic EL results on TAC14eval. Indegree: using indegree of the wikipedia entity as tiebreaker in the improved PageRank algorithm; Cosine: using TFIDF cosine similarity between the wikipedia entry and the source document as tiebreaker in the improved PageRank algorithm; Lucene: using Lucene similarity score as tiebreaker in the improved PageRank algorithm. NYU*n* in parathesis are submission ids that we use in the official TAC evaluation

We used the following external resources for alias generation and nested name extraction: CrossWiki, WikiLinks and a city name gazetteer (cities with population larger than 15,000) provided by `geonames.org`.

8.2 Name Tagging

We extract nested names using heuristic rules described in Section 7.2 and compare the results against a flat name tagger. We report results on TAC14train in Table 9. On this development set, heuristic rules help the overall F-score of name tagging by 0.5 absolute percent: expectedly, the rules boost recall at the cost of precision, but the gazetteer-based rules add more true positive than false positives.

Finally, official results on TAC14eval in Table 10 confirm the trend we see on the development set. The nested tagger using heuristic rules outperforms the flat tagger, although by a smaller margin.

8.3 Linking and Clustering

8.3.1 Linking and Clustering for the EL task

We developed our system using TAC12 as development set and use TAC13 and TAC14train as validation sets. We experiment with the original PageRank algorithm (Original PR) and our improved algorithm(Improved PR). We report re-

	Linking P	Linking R	Linking F1	CEAFm P	CEAFm R	CEAFm F1
Original PageRank	0.589	0.612	0.600	0.555	0.575	0.565
Improved PageRank	0.594	0.618	0.606	0.558	0.578	0.568

Table 13: Linking and clustering results on TAC14train. P: Precision; R: Recall.

	Linking P	Linking R	Linking F1	CEAFm P	CEAFm R	CEAFm F1
NYU1	0.591	0.593	0.592	0.670	0.673	0.671
NYU2	0.587	0.590	0.589	0.669	0.672	0.671
NYU3	0.600	0.583	0.591	0.681	0.660	0.670
NYU4	0.598	0.580	0.589	0.680	0.660	0.670

Table 14: Linking and clustering results on TAC14eval. P: Precision; R: Recall. NYU*n* in parathesis are submission ids that we use in the official TAC evaluation: NYU1 and NYU2 use nested names to genereate queries, while NYU3 and NYU4 use flat names to generated queries. NYU1 and NYU3 use improved PageRank, while NYU2 and NYU4 use original PageRank

sults in Linking F1 and B^3+ F1 in Table 11. Results show that the improved PageRank algorithm steadily obtains 0.7 to 1.6 absolute percent improvement over the original PageRank algorithm.

We further report the official results on TAC2014eval in Table 12. In addition to the two versions of the PageRank algorithm, we experiment with different tie-breakers when the PageRank algorithms produce the same scores for more than one entities: TFIDF cosine similarity between the Wikipedia entry and the source document, similarity score produced by Lucene, and indegree of the entry in Wikipedia. Results show that the improved PageRank algorithm outperforms the original PageRank by 1.3 absolute point, which is consistent with the results on development sets. Various tie-breakers slightly help performance.

8.3.2 Linking and Clustering for the EDL task

In Table 13, we report the Linking F1 score and the mention CEAF scores on TAC14train. Our version of the PageRank algorithm still outperforms the original version, but compared to the diagnostic EL task, which uses perfect mentions, the gap between the two becomes much smaller. The PageRank algorithm and our modification both rely on the hypothesis that correct candidates will help each other. We suspect that the noise introduced by imperfect queries in the EDL task hurts both algorithms and prevents the better algorithm from working as reliably as in the diagnostic EL task.

We see similar results on TAC14eval, reported in Table 14. Our version of the PageRank algorithm still outperforms the original version on Linking performance, but obtains the same CEAF F1 score for clustering as the classical PageRank algorithm.

9 Summary of the Entity Discovery and Linking System

This is the first NYU submission the KBP EL/EDL task, which is a task that involves multiple stages of linguistic processing. We currently follow a pipelined approach, in which query generation, candidate generation, candidate ranking, and NIL clustering all function as separate components. We used variations of the PageRank algorithm to perform collaborative candidate ranking and was able to improve upon the classical PageRank algorithm.

In future submissions, we hope that our downstream components could better utilize information from upstream components. For example, the PageRank candidate ranker could put more weight on the entities that the name tagger has more confidence. We also expect that joint modeling of different stages in the pipeline could further boost EDL performance.

References

Ayman Alhelbawy and Robert Gaizauskas 2014. *Graph Ranking for Collective Named Entity Disambiguation.* In Proceedings of ACL 2014.

Juri Ganitkevitch, Benjamin Van Durme and Chris Callison-burch 2013. *PPDB: The paraphrase database.* In Proceedings of HLT-NAACL 2013.

Ralph Grishman 2013. *Off to a Cold Start: New York Universitys 2013 Knowledge Base Population System*. In Proceedings of TAC 2013.

Andrew McCallum, Dayne Freitag, and Fernando CN Pereira 2000. *Maximum entropy markov models for information extraction and segmentation*. In Proceedings of ICML 2000.

Bonan Min, Xiang Li, Ralph Grishman, and Ang Sun 2012. *New York University 2012 System for KBP Slot Filling*. In Proceedings of TAC 2012.

Dat Ba Nguyen, Johannes Hoffart, Martin Theobald and Gerhard Weikum 2014. *AIDA-light: High-Throughput Named-Entity Disambiguation*. In Proceedings of LDOW 2014.

Maria Pershina, Bonan Min, Wei Xu and Ralph Grishman 2014. *Infusion of Labeled Data into Distant Supervision for Relation Extraction*. In Proceedings of ACL 2014.

Will Radford, Will Cannings, Andrew Naoum, Joel Nothman, Glen Pink, Daniel Tse, and James R Curran 2012. *(Almost) Total Recall - SYDNEY CMCRC at TAC 2012*. In Proceedings of ACL 2014.

Ariel S Schwartz and Marti A Hearst 2002. *A simple algorithm for identifying abbreviation definitions in biomedical text*. In Proceedings of Pacific Symposium on Biocomputing.

Sameer Singh, Amarnag Subramanya, Fernando Pereira, and Andrew McCallum 2012. *Wikilinks: A Large-scale Cross-Document Coreference Corpus Labeled via Links to Wikipedia*. Technical report, University of Massachusetts.

Valentin I Spitkovsky and Angel X Chang 2012. *A Cross-Lingual Dictionary for English Wikipedia Concepts*. In LREC 2012.

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

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati and Christopher D. Manning 2012. *Multi-instance Multi-label Learning for Relation Extraction*. In Proceedings of EMNLP-CoNLL 2012.

Stephen Tratz and Eduard Hovy 2011. *A Fast, Accurate, Non-Projective, Semantically-Enriched Parser*. In Proceedings of EMNLP 2011.