Stanford's Distantly-Supervised Slot-Filling System

Mihai Surdeanu, Sonal Gupta, John Bauer, David McClosky,
Angel X. Chang, Valentin I. Spitkovsky, Christopher D. Manning
Computer Science Department
Stanford University, Stanford, CA 94305
{mihais,sonalg,horatio,mcclosky,angelx,vals,manning}@stanford.edu

Abstract

This paper describes the design and implementation of the slot filling system prepared by Stanford's natural language processing group for the 2011 Knowledge Base Population (KBP) track at the Text Analysis Conference (TAC). Our system relies on a simple distant supervision approach using mainly resources furnished by the track's organizers: we used slot examples from the provided knowledge base, which we mapped to documents from several corpora: those distributed by the organizers, Wikipedia, and web snippets. This system is a descendant of Stanford's system from last year, with several improvements: an inference process that allows for multi-label predictions and uses world-knowledge to validate outputs; model combination; and a tighter integration of entity coreference and web snippets in the training process. Our submissions scored 16 F_1 points using web snippets and 13.5 F_1 without web snippets (both scores are higher than the median score of 12.7 F_1). We also describe our temporal slot filling system, which achieved 37.0 F_1 on the diagnostics temporal task on the developmental queries.

1 Introduction

This paper describes the slot filling system prepared by Stanford's natural language processing (NLP) group for the Knowledge Base Population (KBP) track of the 2011 Text Analysis Conference (TAC). This system is derived from Stanford's distantly-supervised system submitted last year, with several important changes. First, we re-implemented the inference component. The current model allows multiple labels to be assigned to the same slot value. For example, "California" could be extracted as both `per:stateorprovince_of_birth` and `per:stateorprovince_of_residence` for a given entity. Previously, each slot candidate was assigned exactly one label during inference. Furthermore, the inference module now includes a filter that discards slots that do not support several world-knowledge constraints, e.g., that a company cannot be dissolved before it is founded, etc. Second, we implemented a system combination model, which votes between ten different systems trained on different fragments of the knowledge base. Third, we incorporated web snippets and coreference chains (for entity matching) into training. Previously, this information was used only in inference. Lastly, we implemented several extensions to handle the temporal slot filling task. We developed a system that identifies temporal expressions in text (e.g., "first Friday of this month") and normalizes them to the required format. The system then finds temporal constraints for the slot names and values. We found that using simple heuristics and dates of the retrieved documents gives a reasonably high F_1 score. We hope that our current temporal system serves as a "baseline" for future temporal systems.

Using this system we participated in the main and temporal slot filling tasks. In the main task we submitted two runs: one where web snippets were used in both training and evaluation; and one where no web information was used. These runs scored 16 and 13.5 F_1 points, respectively. (Both scores are higher than the median score of all submissions,

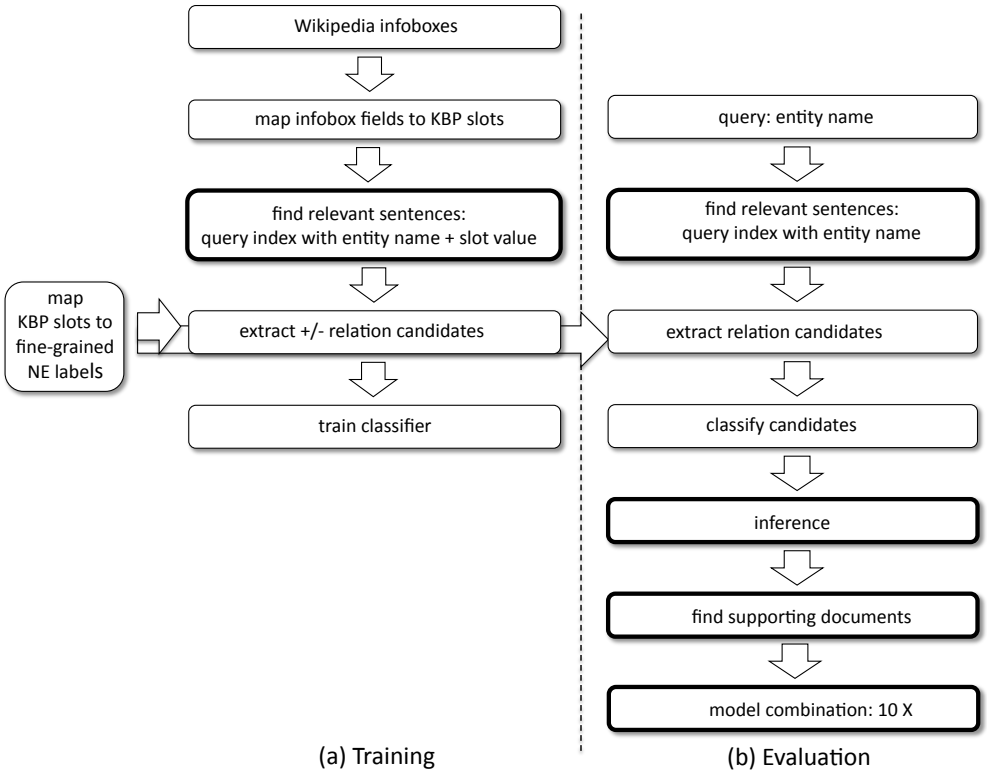

Figure 1: Architecture of the slot filling system. Bolded blocks are either new or significantly changed since last year.

12.7 F_1.) In the temporal task, we submitted our results for two sub-tasks: regular and diagnostics. In the diagnostics temporal task, the system is given the slot names, values and the documents. As we do not know the performance of the system on test queries at the time of writing this paper, we report results on the developmental queries.[1] We achieved 37.0 F_1 on the diagnostics task and 1.7 F_1 on the regular task. The system performs much worse in the regular task because it infers temporal values for slot names and values extracted by the main slot filling system, and hence is restricted by its performance.

2 Architecture of the Slot Filling System

Figure 1 summarizes our system's architecture. For clarity, we present two distinct execution flows: one for training the slot classifier, and one for evaluating the entire system. We describe next all the system components but we focus mostly on the modules that are either new or were significantly changed this year. For details on the other components, we refer the reader to our paper from the 2010 TAC-KBP evaluation (Surdeanu et al., 2010).

2.1 Training

The training process starts by mapping Wikipedia infobox fields to KBP slot types. For example, the infobox field `University:established` maps to the KBP slot type `org:founded`. Next, we retrieve sentences that contain the previously generated slot instances by querying all our document collections with a set of queries, where each query contains an entity name and one slot value, e.g., "Barack Obama" `AND` "United States" for the slot `per:employee_of`. From the documents retrieved, we keep only sentences that contain both the entity name and the slot value. This process differs from last year's system in two ways. First, we retrieve more sentences per entity: for each entity, we retrieve up to 200 sentences per entity from non-web collections and up to 500 sentences per entity from

[1] Note that we do not use any information about the developmental queries during training of the system.

1	Slots defined in groups per:city_of_birth, per:stateorprovince_of_birth and per:country_of_birth must exist and be compatible in a gazetteer of world locations.
2	Slots defined in groups per:city_of_death, per:stateorprovince_of_death and per:country_of_death must exist and be compatible in a gazetteer of world locations.
3	Slots of org:city_of_headquarters, org:stateorprovince_of_headquarters and org:country_of_headquarters must also exist and be compatible in a gazetteer...
4	per:date_of_birth should be before per:date_of_death, if both are defined.
5	org:founded should be before org:dissolved, if both are defined.
6	The set of extracted slot values for org:subsidiaries must be distinct from the set of slot values for org:parents.

Table 1: World-knowledge constraints used during inference.

the collection containing web snippets. This year we could do this without significant overhead because we preprocessed (our preprocessing includes named entity recognition, parsing, and coreference resolution) all our collections offline. Second, we used a different set of document collections, consisting of:

1. The official document corpus provided by the task organizers.

2. Snippets of Wikipedia articles from the 2009 TAC-KBP Evaluation Reference Knowledge Base, preprocessed similarly to last year's system.

3. A complete Wikipedia snapshot from June 2010.

4. A collection of entity-specific web snippets, extracted as follows: for each entity, we constructed a set of queries consisting of the entity name plus one of the trigger phrases from our list of slot-specific triggers.[2] For each query, we retrieved the top ten snippets from Google.

For the extraction of slot candidates, we followed Mintz et al. (2009), i.e., we assumed that all sentences containing a reference to the entity of interest and a known slot value are positive examples for the corresponding slot type. We consider as valid entity references any mentions in a coreference cluster where at least one element matches the entity name. For coreference resolution we used the system of Lee et al. (2011). We considered as negative slot examples all named entity mentions that do not match a known slot value. Additionally, these mentions must appear in the same sentence with the entity whose slots are currently modeled and have a type known to match a KBP slot. We trained the slot classifier using a battery of one-vs-rest logistic regression models, one for each slot type. To control for the excessive number of negative examples, we subsampled them with a probability of 0.01 (Riedel et al., 2010). We used the same features as last year.

[2] http://www.surdeanu.name/mihai/kbp2010/trigger_words.txt

2.2 Evaluation

The sentence retrieval process used during evaluation is similar to the one used when training the model, with two exceptions. First, we retrieve more sentences per entity: up to 500 from the Wikipedia and KBP corpora; and up to 1,000 from the collection of web snippets. Second, the queries used during evaluation contain just the entity name. In contrast to last year, we did not include trigger words in the evaluation queries, since in early experiments this led to a higher recall of the retrieval module (most likely because our list is far from complete).

The inference component was re-architected this year. The new algorithm works as follows:

1. For each tuple (entity name, slot value) we sum the classification probabilities for all instances of this tuple in the data and all valid slot types (i.e., per:* slots for persons and org:* slots for organizations). Note that the resulting scores for a given slot type are no longer probabilities.

2. We discard all predictions with a score below a threshold τ_i, which is tuned using a set of development queries. This naturally mod-

	Multi-label Inference	World Knowledge in Inference	Web Snippets	Model Combination
Experiment 1				
Experiment 2	√			
Experiment 3	√	√		
Experiment 4	√	√	√	
Experiment 5	√	√	√	√
Experiment 6	√	√		√

Table 2: Configuration of experiments. Experiment 5 corresponds to our submission that used web snippets. Experiment 6 is our submission without web information.

els multi-label predictions –i.e., the same (entity name, slot value) tuple may have multiple valid labels– and `list` slots –i.e., for a given (entity name, slot type) the system may output multiple slot values. As an example of multi-label prediction, for a given entity, the slot value "California" may be classified as both `per:stateorprovince_of_birth` and `per:stateorprovince_of_residence`. This inference model is similar to (Hoffmann et al., 2011), but our training is local (i.e., one datum per slot mention), whereas Hoffmann et al. (2011) proposed a joint training process.

3. Lastly, for each entity in the test set, we keep the set of predictions with the highest overall score that satisfy a series of world-knowledge constraints. The complete list of constraints is listed in Table 1.

For the slots produced by the inference process, we identify a supporting document by querying the official index for the entity and its slot value. If a document exists in the official index with both terms in the same sentence, that document is returned as the supporting document. If multiple documents meet this criteria, the one with the terms closest together is returned, with ties broken by the information retrieval (IR) score. (And if no document has the two terms in the same sentence, then simply the document with the highest IR score is chosen.)

The evaluation execution flow concludes with model combination. We observed early in the development of the system that training on more than 10% of the provided knowledge base did not improve performance. To still take advantage of all the available training data, we chose to train ten different models (each using a disjoint ten percent of the knowledge base) and combine their outputs. We implemented a simple combination strategy based on voting, where an extracted slot value is included in the final output if it has been proposed by more than τ_c base models.

3 Architecture of the Temporal Slot Filling System

The temporal system extracts 4-tuples of dates, [T1 T2 T3 T4], for each non-NIL slot value, which is either given (as in the case of diagnostics task) or extracted by the main slotfilling system. We extracted temporal expressions from sentences and normalized them to the YYYYMMDD format. When the year or month was missing from a date t, we converted it to a range, $t\text{-}start$ and $t\text{-}end$. For example, 201110XX converted to 20111001 and 20111031. When a date is fully specified, $t\text{-}start$ and $t\text{-}end$ have the same value. We then assigned T3 as $t\text{-}start$ and T2 as $t\text{-}end$. For sentences from which we did not extract any temporal expression, we used the date of their documents, when available. We submitted another system in which we tried to learn n-grams associated with $start$s and ends of events. For example, 2-grams like "joined on" and "left on" generally mark the starts and ends of events, respectively. To learn these n-grams, we used Freebase to get temporal constraints on relations and found sentences in the corresponding Wikipedia articles that contained those temporal expressions. Whenever a temporal value in Freebase matched a temporal expression from a sentence in Wikipedia, we considered 2-grams and 3-grams around a window of 5 words on each side of the temporal expression. We labeled the n-grams depending on whether the tem-

poral value in the Freebase occurred as $start$ or end. We then weighted the n-grams using a tf-idf-like score, using document frequencies from Google's n-gram corpus (LDC catalog number LDC2006T13). When classifying a temporal expression as $start$ or end during the testing time, we computed Jaccard's coefficient between the class n-grams and the n-grams around the temporal expression, and thresholded the values to assign a particular label. We set the values of T1 and T2 if the label was $start$ and T3 and T4 if the label was end.

4 Experiments

We report first experiments that highlight the contributions of the novel components of our system. For these experiments we used the 100 queries from the 2010 KBP evaluation. We randomly selected 20 questions to tune the two system parameters (τ_i and τ_c) and used the remaining 80 for testing. We devised six different experiments, summarized in Table 2. We report the results of this analysis in Table 3. Note that for this analysis we ignored the extracted supporting document (i.e., the scorer parameter `anydoc` is set to true) for two reasons. First, we wanted to focus on the core components of the system (classification, inference, model combination) instead of the extraction of supporting documents. Second, since the gold answers are based on the submitted runs, they are incomplete with respect to the supporting documents.

The analysis in Table 3 indicates that multi-label inference (Experiment 2) improves the baseline (Experiment 1) by over 1 F_1 point. Note that the baseline model selects exactly one label for each slot candidate (similar to our last year's approach). As expected, the improvement is caused by a significant boost in recall (2.5 absolute percentage points). The world-knowledge constraints (Experiment 3) contribute only 0.3 F_1 points. This modest contribution is explained by the fact that these constraints often take incorrect decisions due to incorrect predictions for the slot values involved. For example, constraint 1 in Table 1 may lead to the removal of up to three slot candidates, even when only one of them is incorrect. On the other hand, the web-snippet collection (Experiment 4) contributes a significant 3.2 F_1 points. This is caused by the fact that the web snippets bring to the table more recent data that is concentrated around the relations of interest (due to the queries that include slot-specific trigger words). As the results indicate, this complements well the static document collections. Lastly, model combination is also successful, leading to an increase of 2 F_1 points when the web snippets are used (Experiment 5) and 2.3 points without web snippets (Experiment 6). This is yet another proof that model combination yields a beneficial regularization effect that is not covered by base models. All in all, the improvements in this year's system led to an increase of 6.6 F_1 points (a 42% relative improvement), compared to last year's system.

	P	R	F_1
Experiment 1	20.5	13.1	15.6
Experiment 2	18.7	15.6	16.7
Experiment 3	19.1	15.9	17.0
Experiment 4	**27.1**	16.4	20.2
Experiment 5	26.3	**19.2**	**22.2**
Experiment 6	21.6	17.4	19.3

Table 3: Development evaluation on 80 queries from the 2010 test set. The other 20 queries were used for parameter tuning. These scores were generated using the official KBP scorer but with the `anydoc` parameter set to true. For all experiments that do not involve model combination (1 through 4) the scores are averages over the ten different models trained on distinct partitions of the knowledge base. For the experiments that use model combination (5 and 6) we scored the combined output of the ten base models.

Table 4 lists our official results in this year's evaluation. We submitted two runs: one which used web snippets during training and testing (equivalent to Experiment 5 in Table 2) and one which did not access the web at any time (equivalent to Experiment 6). Both these runs are above the median submission reported by the evaluation's organizers. We find these results encouraging, especially considering the simplicity of our approach. One troubling observation from these results is that they are significantly lower than our development experiments (e.g., the submission that used web snippets scored 4.2 F_1 points lower than Experiment 5). Of course, these numbers are not directly comparable because they use different evaluation queries, but they are

	P	R	F_1
LDC	86.2	72.6	78.8
Top-1 team	35.0	**25.5**	**29.5**
Top-2 team	**49.2**	12.6	20.0
Stanford with web	17.1	15.0	16.0
Stanford without web	14.1	13.0	13.5
Median team	10.3	16.5	12.7

Table 4: Official results on the 2011 test queries.

	P	R	F_1
All-null Baseline	37.0	12.9	19.2
Document Date	66.3	23.2	34.3
Our system	**71.3**	**24.9**	**37.0**
Our system + Start-End	59.5	20.8	30.8

Table 5: Diagnostic temporal results on 2011 queries. We list a baseline score, which assigns NIL to all temporal slots, and two variants of our heuristics. The scores were not available at submission time.

close: the median submissions had almost identical scores in 2010 and 2011. Considering this, the comparison of Tables 3 and 4 suggests that our component that finds supporting documents — the only difference between the two experiments — requires some further attention.

Table 5 shows precision, recall and F_1 scores for the diagnostic temporal task. We can see that document date alone is very useful in predicting the temporal values. Our system uses both document dates and temporal information in sentences. Using the Start-End system decreased the scores, highlighting the need for more sophisticated approaches.

5 Conclusions and Future Work

This paper describes Stanford's submission to the TAC-KBP 2011 slot filling evaluation. This system extends our distantly-supervised approach from last year with a better inference model, model combination, and access to more data sources. Our results are above the median score (12.7 F_1): 16 F_1 points for the run that accessed the web and 13.5 F_1 for the run without web access. An ablative analysis indicates that the improvements to this year's system are responsible for a 42% performance gain. Our temporal slot filling system indicates that using document dates and simple heuristics give reasonably high F_1 scores.

At least one crucial element is missing from our system: Riedel et al. (2010) showed that the assumption that sentences that contain an entity name and known slot values are positive examples for the relevant slot type is often wrong, especially in non-Wikipedia collections. Several recent models address this problem (Riedel et al., 2010; Hoffmann et al., 2011). We will implement a similar approach in future work. We will also improve our temporal system by using more sophisticated approaches.

References

R. Hoffmann, C. Zhang, X. Ling, L. Zettlemoyer, D.S. Weld. 2011. Knowledge-Based Weak Supervision for Information Extraction of Overlapping Relations. In *ACL*.

H. Lee, Y. Peirsman, A. Chang, N. Chambers, M. Surdeanu, D. Jurafsky. 2011. Stanford's Multi-Pass Sieve Coreference Resolution System at the CoNLL-2011 Shared Task. In *Proceedings of the CoNLL-2011 Shared Task*.

M. Mintz, S. Bills, R. Snow, and D. Jurafsky. 2009. Distant supervision for relation extraction without labeled data In *ACL-IJCNLP*.

S. Riedel, L. Yao, and A. McCallum. 2010. Modeling relations and their mentions without labeled text. In *ECML/PKDD*.

M. Surdeanu, D. McClosky, J. Tibshirani, J. Bauer, A.X. Chang, V.I. Spitkovsky, and C.D. Manning. 2010. A Simple Distant Supervision Approach for the TAC-KBP Slot Filling Task. In *TAC-KBP*.