# 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 track at the Text Analysis Conference. 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 entry into last year's evaluation, 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 submission 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 62 $F_1$ points in the diagnostic temporal task on the test 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 used SUTIME (Chang and Manning, 2012) to identify temporal expressions in text (e.g., "first Friday of this month") and normalize them to the required format. Our system then associated temporal constraints to slot names and values. We found that using simple heuristics — and anchoring them to the dates of the retrieved documents — can already yield reasonably high $F_1$ scores.

We entered our system into both 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 of all entries, 12.7 $F_1$.) In the temporal task, we submitted our results for two sub-tasks: regular and diagnostic. (In the diagnostic temporal task, the system is given correct slot names, values and the documents.) We achieved around 62 $F_1$ points in the diagnostic task and 3.7 $F_1$ points in the regular task.

## 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. Next, we describe all of the

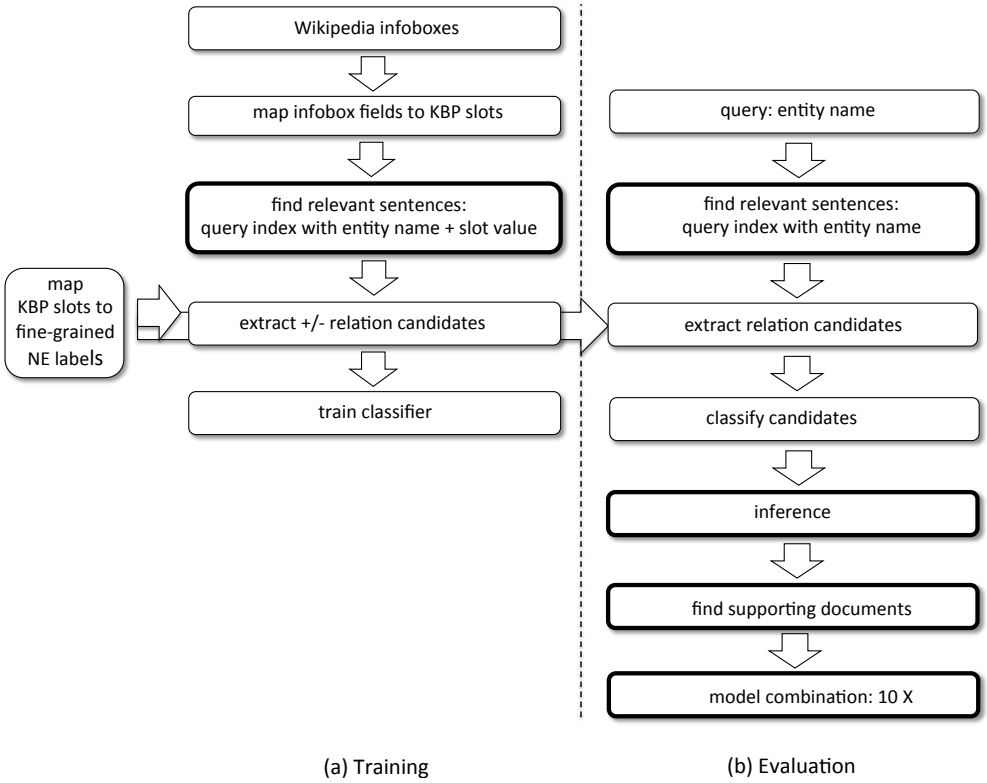


Figure 1: Architecture of the slot-filling system. Bolded blocks are either new or significantly changed since last year.

system components, focusing primarily on ones that are new and those which have significantly changed since last year. For details of the remaining components, we refer the reader to our paper from the previous 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 content from the document collections: for each entity, we retrieve up to 200 sentences from non-web collections and up to 500, per entity, from the index containing web snippets. This year we could do this without significant overhead because we pre-processed (our pre-processing includes named entity recognition, parsing, and coreference resolution) all collections offline. Second, we used a different set of document collections, consisting of:

1. The official corpus provided by the organizers;
2. Snippets of Wikipedia articles from the 2009 KBP Evaluation Reference Knowledge Base, pre-processed similarly to last year's system;
3. An English Wikipedia dump 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.[1] For each query, we retrieved the top ten snippets from Google.

[1] `http://nlp.stanford.edu/pubs/kbp_trigger_words.txt`

| | |
|---|---|
| 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 | Extracted slot values for org:subsidiaries must not overlap with the set for org:parents. |

Table 1: World-knowledge constraints used during inference.

To extract slot candidates, we followed Mintz et al. (2009) in assuming 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 considered as "valid references" any mentions from coreference clusters where at least one element matched the entity's name, using the coreference resolution system of Lee et al. (2011).

We generated negative slot examples as follows:

- We extracted candidates: all named entity mentions in a window of 20 words to the left and right of a valid mention of the entity of interest;
- We compared each candidate with the infobox slot values of the given entity. If the candidate's named entity (NE) label is compatible with a slot type (e.g., the NE label LOC is compatible with per:stateorprovince_of_birth) and its text does not match the corresponding slot value in the infobox, then we mark this candidate as a negative example for that slot type.

Note that this heuristic yields a different set of negative examples for each slot type. To accommodate this setup, we trained the slot classifier using a battery of one-versus-rest logistic regression models, one for each slot type (rather than a single multi-class classifier, which would require a single set of negative examples). We used the same features as last year. To control for the excessive number of negative examples, we subsampled them with a probability of 0.01 (Riedel et al., 2010).

### 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. (In preliminary experiments, this led to a higher recall of the retrieval module — most likely because our list is far from complete.)

We re-architected the inference component this year. Our 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 $\tau_i$, which is tuned using a set of development queries. This naturally models multi-label predictions, where the same (entity name, slot value) tuple may have multiple valid labels. 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. Our inference model is similar to that of Hoffmann et al. (2011), but with local training (i.e., one datum per slot mention, instead of having all mentions with the same value modeled jointly).
3. Lastly, for each entity in the test data, we keep the set of predictions with the highest overall score that satisfy a series of world-knowledge constraints (see Table 1 for the complete list).

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 criterion, the one with the terms closest together is returned, with ties broken by the information retrieval (IR) score. (And if there is no such document, we fall back to the highest-IR result.)

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 $\tau_c$ base models.

## 3 Architecture for Temporal Slot-Filling

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 diagnostic task) or extracted by the main slot-filling 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 is 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 expressions, we used the date of their documents, when available.

### 3.1 A Start/End Model

We also submitted a second system in which we tried to learn $n$-grams associated with $start$s and $end$s of events. For example, bigrams 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 bigrams and trigrams around a window of five words on each side of the temporal expression. We labeled the $n$-grams depending on whether the temporal 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 $end$ at 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 ($\tau_i$ and $\tau_c$) and used the remaining 80 for testing. We devised six different experiments (summarized in Table 2); Table 3 shows the results. Note that in this analysis we ignored the extracted supporting document (i.e., scorer parameter `anydoc` was set to true), for two reasons. First, we wanted to focus on the core components of the system (classification, inference, and model combination), instead of the extraction of supporting documents. And second, since gold answers are based on submitted runs, they are incomplete with respect to supporting documents.

Table 3 indicates that multi-label inference (Experiment 2) improves over the baseline (Experiment 1) by more than 1 $F_1$ point. Note that the baseline model selects exactly one label for each slot candidate (similarly to our approach from last year). 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 improvement can be explained by constraints often *emphasizing* the errors made by the local classifier, instead of correcting them. For example, constraint 1 in Table 1 may lead to the removal of up to three slot

| | 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: #5 is our submission with web snippets; and #6 is our system without them.

| | 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; remaining 20 queries were used for parameter tuning. These scores were generated using the official KBP scorer, with the `anydoc` parameter set to true. For experiments without model combination (1–4), the scores are averages over the ten different models trained on distinct partitions of the knowledge base; for the other experiments (5–6), which do involve model combination, we scored the combined output of the ten base models.

candidates, even when only one of them is incorrect.

On the other hand, web-snippets (Experiment 4) contribute a significant 3.2 $F_1$ points, most likely because they bring to the table more recent data that was already deemed as highly correlated with the entity of interest by the search engine. Our results indicate that such dynamically-added information is complementary to static collections of documents.

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 further evidence that model combination yields a beneficial regularization effect that is not provided by base models alone. All in all, this year's improvements led to an increase of 6.6 $F_1$ points (a 42% relative improvement), compared to last year's system.

Table 4 lists our official scores in this year's evaluation. We submitted two runs: one using web snippets in training and testing (equivalent to Experiment 5 in Table 2) and one which did not access the web at all (equivalent to Experiment 6). Both runs scored above the median entry reported by the evaluation's organizers. We find this outcome encouraging, especially considering the simplicity of our approach. One troubling observation, however, is that our entries scored significantly lower than in development (e.g., the $F_1$ for our submission using web snippets is 4.2 points below Experiment 5). Although these numbers are not directly comparable (because of differences in evaluation queries), this large difference suggests that our component for finding supporting documents — the only difference between the two set-ups other then the evaluation queries — requires further attention.

| | 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.

### 4.1 Temporal Slot-Filling

Our submission attained around 62 $F_1$ on the diagnostic task, but only 3.7 $F_1$ points for the full temporal slot-filling task, on the test queries. The best system scored around 64 $F_1$ for the diagnostic and 22.7 $F_1$ on the full task. It is unclear why our system performs poorly on the full task, but it is likely that there may have been an integration bug between the temporal module and the slot-filling system.

Table 5 shows precision, recall and $F_1$ scores for the diagnostic temporal task on the development queries. The document date alone is already quite predictive of temporal values. Our system improves over this baseline by using both document dates and temporal information in sentences. However, adding also the Start/End model had a negative impact, signaling a need for more sophisticated approaches.

## 5 Conclusions and Future Work

This paper describes Stanford's submission to the TAC-KBP 2011 slot-filling evaluation. Our system extends the distantly-supervised approach from last year (Surdeanu et al., 2010) with a better inference model, model combination, and access to more data sources. We attained results that 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% (relative) performance gain. In temporal slot-filling, we showed that using document dates and simple heuristics gives reasonably high $F_1$ scores.

Our results in the main slot-filling task are close to the distant supervision component of the top-performing system (Sun et al., 2011). That combination of techniques (distant supervision, rule-based, and question answering) obtained a much better overall score. But the fact that two independent groups obtained similar scores with distant supervision suggests that we may be approaching its ceiling.

There is at least one crucial element missing from our system: Riedel et al. (2010) showed that the assumption that sentences containing an entity name and known slot values are positive examples for the relevant slot type is often wrong — particularly in non-Wikipedia collections; several recently-proposed models (Riedel et al., 2010; Hoffmann et al., 2011, *inter alia*) address this problem. In the future, we will work on relaxing our dependence on this strong assumption. And we will also explore ways of improving our temporal slot-filling system.

| | P | R | $F_1$ |
|---|---|---|---|
| All-null baseline | 37.0 | 12.9 | 19.2 |
| Document-date baseline | 66.3 | 23.2 | 34.3 |
| Our system | **71.3** | **24.9** | **37.0** |
| Our system + Start/End model | 59.5 | 20.8 | 30.8 |

Table 5: Diagnostic temporal results on the 2011 development queries. We built two baselines: the first sets all temporal slots to NIL; the second assigns T2 and T3 based on the document date. Our system, which extends the second baseline by using the date syntactically modifying the slot, when available, is listed next. The last row shows our system combined with a language model for recognizing starts and ends of events, based on Freebase.

## Acknowledgments

We thank the task organizers for their effort.

This material is based upon work supported by the Air Force Research Laboratory (AFRL) under prime contract no. FA8750-09-C-0181. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the Air Force Research Laboratory. Angel X. Chang is supported by a Stanford Graduate Fellowship.

## References

A.X. Chang and C.D. Manning. 2012. SUTIME: A Library for Recognizing and Normalizing Time Expressions. In *LREC*.

R. Hoffmann, C. Zhang, X. Ling, L. Zettlemoyer, and 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, and 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*.

A. Sun, R. Grishman, W. Xu, and B. Min. 2011. New York University 2011 System for KBP Slot Filling. In *TAC*.

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*.