In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL-15), pp. 177-187, Beijing, China, July 2015.

Stacked Ensembles of Information Extractors for Knowledge-Base Population

Nazneen Fatema Rajani* **Vidhoon Viswanathan*** **Yinon Bentor** **Raymond J. Mooney**
Department of Computer Science
University of Texas at Austin
Austin, TX 78712, USA
{nrajani,vidhoon,yinon,mooney}@cs.utexas.edu

Abstract

We present results on using stacking to ensemble multiple systems for the Knowledge Base Population English Slot Filling (KBP-ESF) task. In addition to using the output and confidence of each system as input to the stacked classifier, we also use features capturing how well the systems agree about the provenance of the information they extract. We demonstrate that our stacking approach outperforms the best system from the 2014 KBP-ESF competition as well as alternative ensembling methods employed in the 2014 KBP Slot Filler Validation task and several other ensembling baselines. Additionally, we demonstrate that including provenance information further increases the performance of stacking.

1 Introduction

Using *ensembles* of multiple systems is a standard approach to improving accuracy in machine learning (Dietterich, 2000). Ensembles have been applied to a wide variety of problems in natural language processing, including parsing (Henderson and Brill, 1999), word sense disambiguation (Pedersen, 2000), and sentiment analysis (Whitehead and Yaeger, 2010). This paper presents a detailed study of ensembling methods for the TAC *Knowledge Base Population* (KBP) *English Slot Filling* (ESF) task (Surdeanu, 2013; Surdeanu and Ji, 2014).

We demonstrate new state-of-the-art results on this KBP task using *stacking* (Wolpert, 1992), which trains a final classifier to optimally combine the results of multiple systems. We present results for stacking all systems that competed in both the 2013 and 2014 KBP-ESF tracks, training on 2013 data and testing on 2014 data. The resulting stacked ensemble outperforms all systems in the 2014 competition, obtaining an F1 of 48.6% compared to 39.5% for the best performing system in the most recent competition.

Although the associated KBP Slot Filler Validation (SFV) Track (Wang et al., 2013; Yu et al., 2014; Sammons et al., 2014) is officially focused on improving the precision of individual existing systems by filtering their results, frequently participants in this track also combine the results of multiple systems and also report increased recall through this use of ensembling. However, SFV participants have not employed stacking, and we demonstrate that our stacking approach outperforms existing published SFV ensembling systems.

KBP ESF systems must also provide *provenance* information, i.e. each extracted slot-filler must include a pointer to a document passage that supports it (Surdeanu and Ji, 2014). Some SFV systems have used this provenance information to help filter and combine extractions (Sammons et al., 2014). Therefore, we also explored enhancing our stacking approach by including additional input features that capture provenance information. By including features that quantify how much the ensembled systems agree on provenance, we further improved our F1 score for the 2014 ESF task to 50.1%.

The remainder of the paper is organized as follows. Section 2 provides background information on existing KBP-ESF systems and stacking. Section 3 provides general background on the KBP-ESF task. Section 4 describes our stacking approach, including how provenance information is used. Section 5 presents comprehensive experiments comparing this approach to existing results and several additional baselines, demonstrating new state-of-the-art results on KBP-ESF. Section 6 reviews prior related work on ensembling

* These authors contributed equally

Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, pages 177–187, Beijing, China, July 26-31, 2015.

for information extraction. Section 7 presents our final conclusions and proposed directions for future research.

2 Background

For the past few years, NIST has been conducting the English Slot Filling (ESF) Task in the Knowledge Base Population (KBP) track among various other tasks as a part of the Text Analysis Conference(TAC)(Surdeanu, 2013; Surdeanu and Ji, 2014). In the ESF task, the goal is to fill specific slots of information for a given set of query entities (people or organizations) based on a supplied text corpus. The participating systems employ a variety of techniques in different stages of the slot filling pipeline, such as entity search, relevant document extraction, relation modeling and inference. In 2014, the top performing system, *DeepDive with Expert Advice* from Stanford University (Wazalwar et al., 2014), employed distant supervision (Mintz et al., 2009) and Markov Logic Networks (Domingos et al., 2008) in their learning and inferencing system. Another system, *RPI_BLENDER* (Hong et al., 2014), used a restricted fuzzy matching technique in a framework that learned event triggers and employed them to extract relations from documents.

Given the diverse set of slot-filling systems available, it is interesting to explore methods for ensembling these systems. In this regard, TAC also conducts a Slot Filler Validation (SFV) task who goal is to improve the slot-filling performance using the output of existing systems. The input for this task is the set of outputs from all slot-filling systems and the expected output is a filtered set of slot fills. As with the ESF task, participating systems employ a variety of techniques to perform validation. For instance, *RPI_BLENDER* used a Multi-dimensional Truth Finding model (Yu et al., 2014) which is an unsupervised validation approach based on computing multidimensional credibility scores. The *UI_CCG* system (Sammons et al., 2014) developed two different validation systems using entailment and majority voting.

However, *stacking* (Sigletos et al., 2005; Wolpert, 1992) has not previously been employed for ensembling KBP-ESF systems. In stacking, a meta-classifier is learned from the output of multiple underlying systems. In our work, we translate this to the context of ensembling slot filling systems and build a *stacked meta-classifier* that learns to combine the results from individual slot filling systems. We detail our stacking approach for ensembling existing slot filling systems in Section 4.

3 Overview of KBP Slot Filling Task

The goal of the TAC KBP-ESF task (Surdeanu, 2013; Surdeanu and Ji, 2014) is to collect information (fills) about specific attributes (slots) for a set of entities (queries) from a given corpus. The queries vary in each year of the task and can be either a person (PER) or an organization (ORG) entity. The slots are fixed and are listed in Table 1 by entity type. Some slots (like per:age) are *single-valued* while others (like per:children) are *list-valued* i.e., they can take multiple slot fillers.

3.1 Input and Output

The input for the task is a set of queries and the corpus in which to look for information. The queries are provided in an XML format containing basic information including an ID for the query, the name of the entity, and the type of entity (PER or ORG). The corpus consists of documents format from discussion forums, newswire and the Internet. Each document is identified by a unique document ID.

The output for the task is a set of slot fills for each input query. Depending on the type, each query should have a *NIL* or one or more lines of output for each of the corresponding slots. The output line for each slot fill contains the fields shown in Table 2. The query ID in Column 1 should match the ID of the query given as input. The slot name (Column 2) is one of the slots listed in Table 1 based on entity type. Run ID (Column 3) is a unique identifier for each system. Column 4 contains a *NIL* filler if the system could not find any relevant slot filler. Otherwise, it contains the *relation provenance*. Provenance is of the form *docid:startoffset-endoffset*, where *docid* specifies a source document from the corpus and the offsets demarcate the text in this document supporting the relation. The offsets correspond to the spans of the candidate document that describe the relation between the query entity and the extracted slot filler. Column 5 contains the extracted slot filler. Column 6 is a filler provenance that is similar in format to relation provenance but in this case the offset corresponds to the portion of the document containing the extracted filler. Column 7 is a confi-

Person		Organization	
per:alternate_names	per:cause_of_death	org:country_of_headquarters	org:founded_by
per:date_of_birth	per:countries_of_residence	org:stateorprovince_of_headquarters	org:date_dissolved
per:age	per:statesorprovinces_of_residence	org:city_of_headquarters	org:website
per:parents	per:cities_of_residence	org:shareholders	org:date_founded
per:spouse	per:schools_attended	org:top_members_employees	org:members
per:city_of_birth	per:city_of_death	org:political_religious_affiliation	org:member_of
per:origin	per:stateorprovince_of_death	org:number_of_employees_members	org:subsidiaries
per:other_family	per:country_of_death	org:alternate_names	org:parents
per:title	per:employee_or_member_of		
per:religion	per:stateorprovince_of_birth		
per:children	per:country_of_birth		
per:siblings	per:date_of_death		
per:charges			

Table 1: Slots for PER and ORG queries

dence score which systems can provide to indicate their certainty in the extracted information.

3.2 Scoring

The scoring for the ESF task is carried out as follows. The responses from all slot-filling systems are pooled and a *key file* is generated by having human assessors judge the correctness of these responses. In addition, LDC includes a manual key of fillers that were determined by human judges. Using the union of these keys as the gold standard, precision, recall, and F1 scores are computed.

Column	Field Description
Column 1	Query ID
Column 2	Slot name
Column 3	Run ID
Column 4	NIL or Relation Provenance
Column 5	Slot filler
Column 6	Filler Provenance
Column 7	Confidence score

Table 2: SF Output line fields

4 Ensembling Slot-Filling Systems

Given a set of query entities and a fixed set of slots, the goal of ensembling is to effectively combine the output of different slot-filling systems. The input to the ensembling system is the output of individual systems (in the format described in previous section) containing slot fillers and additional information such as provenance and confidence scores. The output of the ensembling system is similar to the output of an individual system, but it productively aggregates the slot fillers from different systems.

4.1 Algorithm

This section describes our ensembling approach which trains a final binary classifier using features that help judge the reliability and thus correctness of individual slot fills. In a final *post-processing* step, the slot fills that get classified as "correct" by the classifier are kept while the others are set to NIL.

4.1.1 Stacking

Stacking is a popular ensembling method in machine learning (Wolpert, 1992) and has been successfully used in many applications including the top performing systems in the Netflix competition (Sill et al., 2009). The idea is to employ multiple learners and combine their predictions by training a "meta-classifier" to weight and combine multiple models using their confidence scores as features. By training on a set of supervised data that is disjoint from that used to train the individual models, it learns how to combine their results into an improved ensemble model. We employ a single classifier to train and test on all slot types using an L1-regularized SVM with a linear kernel (Fan et al., 2008).

4.1.2 Using Provenance

As discussed above, each system provides provenance information for every non-NIL slot filler. There are two kinds of provenance provided: the relation provenance and the filler provenance. In our algorithm, we only use the filler provenance for a given slot fill. This is because of the changes in the output formats for the ESF task from 2013 to 2014. Specifically, the 2013 specification requires separate entity and justification provenance fields, but the 2014 collapses these into a single relation provenance field. An additional filler provenance

field is common to both specifications. Hence, we use the filler provenance that is common between 2013 and 2014 formats. As described earlier, every provenance has a *docid* and *startoffset-endoffset* that gives information about the document and offset in the document from where the slot fill has been extracted. The UI-CCG SFV system Sammons et al. (2014) effectively used this provenance information to help validate and filter slot fillers. This motivated us to use provenance in our stacking approach as additional features as input to the meta-classifier.

We use provenance in two ways, first using the *docid* information, and second using the *offset* information. We use the *docids* to define a document-based provenance score in the following way: for a given query and slot, if N systems provide answers and a maximum of n of those systems give the same *docid* in their filler provenance, then the document provenance score for those n slot fills is n/N. Similarly, other slot fills are given lower scores based on the fraction of systems whose provenance document agree with theirs. Since this provenance score is weighted by the number of systems that refer to the same provenance, it measures the reliability of a slot fill based on the document from where it was extracted.

Our second provenance measure uses *offsets*. The degree of overlap among the various systems' offsets can also be a good indicator of the reliability of the slot fill. The Jaccard similarity coefficient is a statistical measure of similarity between sets and is thus useful in measuring the degree of overlap among the offsets of systems. Slot fills have variable lengths and thus the provenance offset ranges are variable too. A metric such as the Jaccard coefficient captures the overlapping offsets along with normalizing based on the union and thus resolving the problem with variable offset ranges. For a given query and slot, if N systems that attempt to fill it have the same *docid* in their document provenance, then the offset provenance (OP) score for a slot fill by a system x is calculated as follows:

$$OP(x) = \frac{1}{|N|} \times \sum_{i \in N, i \neq x} \frac{|\mathsf{offsets(i)} \cap \mathsf{offsets(x)}|}{|\mathsf{offsets(i)} \cup \mathsf{offsets(x)}|}$$

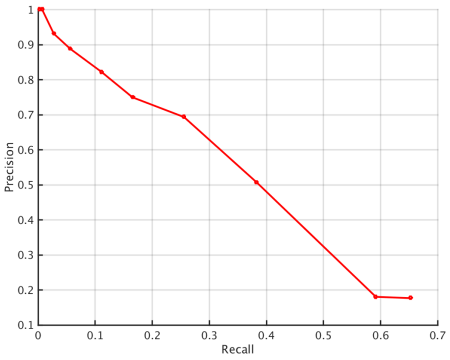
Per our definition, systems that extract slot fills from *different* documents for the same query slot have zero overlap among offsets. We note that the offset provenance is always used along with the document provenance and thus useful in discriminating slot fills extracted from a different document for the same query slot. Like the document provenance score, the offset provenance score is also a weighted feature and is a measure of reliability of a slot fill based on the offsets in the document from where it is extracted. Unlike past SFV systems that use provenance for validation, our approach does not need access to the large corpus of documents from where the slot fills are extracted and is thus very computationally inexpensive.

4.2 Eliminating Slot-Filler Aliases

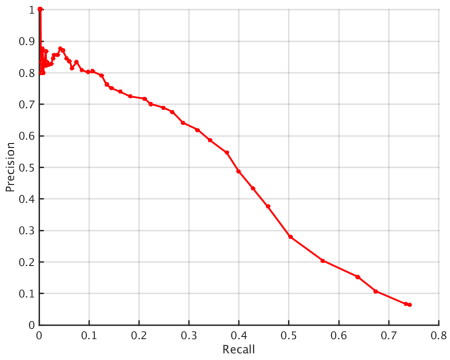
When combining the output of different ESF systems, it is possible that some slot-filler entities might overlap with each other. An ESF system could extract a filler F_1 for a slot *S* while another ESF system extracts another filler F_2 for the same slot *S*. If the extracted fillers F_1 and F_2 are *aliases* (i.e. different names for the same entity), the scoring system for the TAC KBP SF task considers them redundant and penalizes the precision of the system.

In order to eliminate aliases from the output of ensembled system, we employ a technique derived by inverting the scheme used by the LSV ESF system (Roth et al., 2013) for query expansion. LSV ESF uses a Wikipedia anchor-text model (Roth and Klakow, 2010) to generate aliases for given query entities. By including aliases for query names, the ESF system increase the number of candidate sentences fetched for the query.

To eliminate filler aliases, we apply the same technique to generate aliases for all slot fillers of a given query and slot type. Given a slot filler, we obtain the Wikipedia page that is most likely linked to the filler text. Then, we obtain the anchor texts and their respective counts from all other Wikipedia pages that link to this page. Using these counts, we choose top N (we use N=10 as in LSV) and pick the corresponding anchor texts as aliases for the given slot filler. Using the generated aliases, we then verify if any of the slot fillers are redundant with respect to these aliases. This scheme is not applicable to slot types whose fillers are not entities (like date or age). Therefore, simpler matching schemes are used to eliminate redundancies for these slot types.



Common systems dataset



All 2014 SFV systems dataset

Figure 1: Precision-Recall curves for identifying the best voting performance on the two datasets

5 Experimental Evaluation

This section describes a comprehensive set of experiments evaluating ensembling for the KBP ESF task. Our experiments are divided into two subsets based on the datasets they employ. Since our stacking approach relies on 2013 SFV data for training, we build a dataset of one run for every team that participated in *both* the 2013 and 2014 competitions and call it the *common systems dataset*. There are 10 common teams of the 17 teams that participated in ESF 2014. The other dataset comprises of all 2014 SFV systems (including all runs of all 17 teams that participated in 2014). There are 10 systems in the common systems dataset, while there are 65 systems in the all 2014 SFV dataset. Table 3 gives a list of the common systems for 2013 and 2014 ESF task. ESF systems do change from year to year and it's not a perfect comparison, but systems generally get better every year and thus we are probably only underperforming.

Common Systems
LSV
IIRG
UMass_IESL
Stanford
BUPT_PRIS
RPI_BLENDER
CMUML
NYU
Compreno
UWashington

Table 3: Common teams for 2013 and 2014 ESF

5.1 Methodology and Results

For our unsupervised ensembling baselines, we evaluate on both the common systems dataset as well as the entire 2014 SFV dataset. We compare our stacking approach to three unsupervised baselines. The first is *Union* which takes the combination of values for all systems to maximize recall. If the slot type is list-valued, it classifies all slot fillers as correct and always includes them. If the slot type is single-valued, if only one systems attempts to answer it, then it includes that system's slot fill. Otherwise if multiple systems produce a response, it only includes the slot fill with the highest confidence value as correct and discards the rest.

The second baseline is *Voting*. For this approach, we vary the threshold on the number of systems that must agree on a slot fill from one to all. This gradually changes the system from the union to intersection of the slot fills, and we identify the threshold that results in the highest F1 score. We learn a threshold on the 2013 SFV dataset (containing 52 systems) that results in the best F1 score. We use this threshold for the voting baseline on 2014 SFV dataset. As we did for the 2013 common systems dataset, we learn a threshold on the 2013 common systems that results in the best F1 score and use this threshold for the voting baseline on 2014 common systems.

The third baseline is an "oracle threshold" version of *Voting*. Since the best threshold for 2013 may not necessarily be the best threshold for 2014, we identify the best threshold for 2014 by plotting a Precision-Recall curve and finding the best F1 score for the voting baseline on both the SFV and common systems datasets. Figure 1 shows the

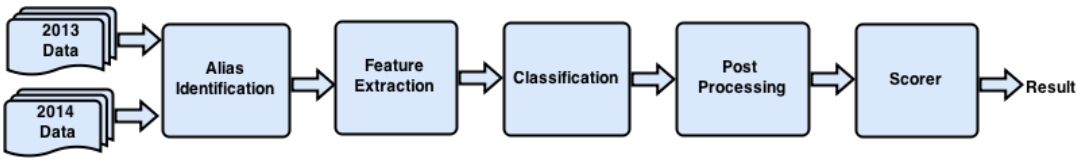


Figure 2: Our system pipeline for evaluating supervised ensembling approaches

Baseline	Precision	Recall	F1
Union	0.067	**0.762**	0.122
Voting (threshold learned on 2013 data)	**0.641**	0.288	0.397
Voting (optimal threshold for 2014 data)	0.547	0.376	**0.445**

Table 4: Performance of baselines on all 2014 SFV dataset (65 systems)

Approach	Precision	Recall	F1
Union	0.176	**0.647**	0.277
Voting (threshold learned on 2013 data)	**0.694**	0.256	0.374
Best ESF system in 2014 (Stanford)	0.585	0.298	0.395
Voting (optimal threshold for 2014 data)	0.507	0.383	0.436
Stacking	0.606	0.402	0.483
Stacking + Relation	0.607	0.406	0.486
Stacking + Provenance (document)	0.499	0.486	0.492
Stacking + Provenance (document) + Relation	0.653	0.400	0.496
Stacking + Provenance (document and offset) + Relation	0.541	0.466	**0.501**

Table 5: Performance on the common systems dataset (10 systems) for various configurations. All approaches except the Stanford system are our implementations.

Precision-Recall curve for two datasets for finding the best possible F1 score using the voting baseline. We find that for the common systems dataset, a threshold of 3 (of 10) systems gives the best F1 score, while for the entire 2014 SFV dataset, a threshold of 10 (of 65) systems gives the highest F1. Note that this gives an upper bound on the best results that can be achieved with voting, assuming an optimal threshold is chosen. Since the upper bound can not be predicted without using the 2014 dataset, this baseline has an unfair advantage. Table 4 shows the performance of all 3 baselines on the all 2014 SFV systems dataset.

For all our supervised ensembling approaches, we train on the 2013 SFV data and test on the 2014 data for the common systems. We have 5 different supervised approaches. Our first approach is stacking the common systems using their confidence scores to learn a classifier. As discussed earlier, in stacking we train a meta-classifier that combines the systems using their confidence scores as features. Since the common systems dataset has 10 systems, this classifier uses 10 features. The second approach also provides stacking with a nominal feature giving the relation name (as listed in Table 1) for the given slot instance. This allows the system to learn different evidence-combining functions for different slot types if the classifier finds this useful. For our third approach, we also provide the document provenance feature described in Section 4.1. Altogether this approach has 11 features (10 confidence score + 1 document provenance score). The fourth approach uses confidences, the document provenance feature, and a one-hot encoding of the relation name for the slot instance. Our final approach also includes the offset provenance (OP) feature discussed in Section 4.1. There are altogether 13 features in this approach. All our supervised approaches use the Weka package (Hall et al., 2009) for training the meta-classifier, using an L1-regularized SVM with a linear kernel (other classifiers gave similar results). Figure 2 shows our system pipeline for evaluating supervised ensembling approaches. Table 5 gives the performance of all our supervised approaches as well as

our unsupervised baselines for the common systems dataset.

Analysis by Surdeanu and Ji (2014) suggests that 2014 ESF queries are more difficult than those for 2013. They compare two systems by running both on 2013 and 2014 data and find there is a considerable drop in the performance of both the systems. We note that they run the same exact system on 2013 and 2014 data. Thus, in order to have a better understanding of our results, we plot a learning curve by training on different sizes of the 2013 SFV data and using the scorer to measure the F1 score on the 2014 SFV data for the 10 common systems. Figure 3 shows the learning curve thus obtained. Although there are certain parts of the dataset when the F1 score drops which we suspect is due to overfitting the 2013 data, there is still a strong correlation between the 2013 training data size and F1 score on the 2014 dataset. Thus we can infer that training on 2013 data is useful even though the 2013 and 2014 data are fairly different. Although the queries change, the common systems remain more-or-less the same and stacking enables a meta-classifier to weigh those common systems based on their 2013 performance.

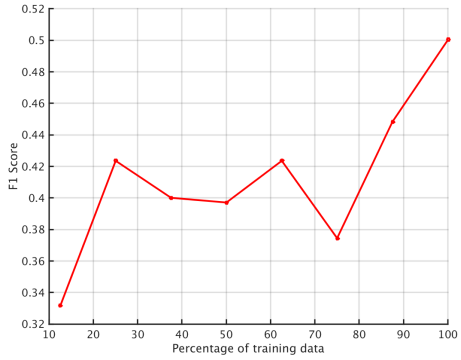


Figure 3: Learning curve for training on 2013 and testing on 2014 common systems dataset

To further validate our approach, we divide the 2013 SFV data based on the systems that extracted those slot fills. Then we sort the systems, from higher to lower, based on the number of false positives produced by them in the ensembling approach. Next we train a classifier in an incremental fashion adding one system's slot fills for training at each step and analyzing the performance on 2014 data. This allows us to analyze the results at the system level. Figure 4 shows the plot of F1 score vs. the number of systems at each step. The figure shows huge improvement in F1 score at steps 6 and 7. At step 6 the Stanford system is added to the pool of systems which is the best performing ESF system in 2014 and fourth best in 2013. At step 7, the UMass system is added to the pool and, although the system on it own is weak, it boosts the performance of our ensembling approach. This is because the UMass system alone contributes approximately 24% of the 2013 training data (Singh et al., 2013). Thus adding this one system significantly improves the training step leading to better performance. We also notice that our system becomes less conservative at this step and has higher recall. The reason for this is that the systems from 1 to 5 had very high precision and low recall whereas from system 6 onwards the systems have high recall. Thus adding the UMass system enables our meta-classifier to have a higher recall for small decrease in precision and thus boosting the overall F1 measure. Without it, the classifier produces high precision but low recall and decreases the overall F1 score by approximately 6 points.

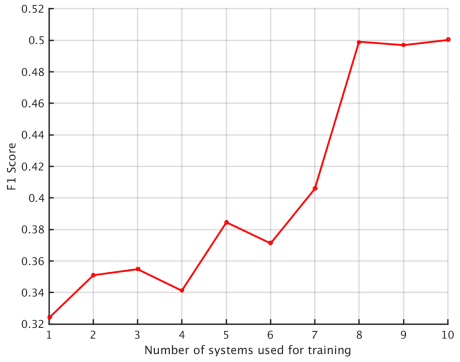


Figure 4: Incrementally training on 2013 by adding a system at each step and testing on 2014 common systems dataset

We also experimented with cross validation within the 2014 dataset. Since we used only 2014 data for this experiment, we also included the relation provenance as discussed in Section 4.1.2. Table 6 shows the results on 10-fold cross-validation on 2014 data with only the filler provenance and with both the filler and relation provenance. The performance of using only the filler provenance is slightly worse than training on 2013 because the 2014 SFV data has many fewer instances but uses more systems for learning compared to the 2013

Approach	Precision	Recall	F1
Stacking + Filler provenance + Relation	0.606	0.415	0.493
Stacking + Filler and Relation provenance + Relation	**0.609**	**0.434**	**0.506**

Table 6: 10-fold Cross-Validation on 2014 SFV dataset (65 systems)

Baseline	Precision	Recall	F1
Union	0.054	**0.877**	0.101
Voting (threshold learned on 2013 data)	**0.637**	0.406	0.496
Voting (optimal threshold for 2014 data)	0.539	0.526	**0.533**

Table 7: Baseline performance on all 2014 SFV dataset (65 systems) using unofficial scorer

Approach	Precision	Recall	F1
Union	0.177	**0.922**	0.296
Voting (threshold learned on 2013 data)	**0.694**	0.256	0.374
Best published SFV result in 2014 (UIUC)	0.457	0.507	0.481
Voting (optimal threshold for 2014 data)	0.507	0.543	0.525
Stacking + Provenance(document)	0.498	0.688	0.578
Stacking	0.613	0.562	0.586
Stacking + Relation	0.613	0.567	0.589
Stacking + Provenance (document and offset) + Relation	0.541	0.661	0.595
Stacking + Provenance (document) + Relation	0.659	0.56	**0.606**

Table 8: Performance on the common systems dataset (10 systems) for various configurations using the unofficial scorer. All approaches except the UIUC system are our implementations.

SFV data.

The TAC KBP official scoring key for the ESF task includes human annotated slot fills along with the pooled slot fills obtained by all participating systems. However, Sammons et al. (2014) use an unofficial scoring key in their paper that does not include human annotated slot fills. In order to compare to their results, we also present results using the same unofficial key. Table 7 gives the performance of our baseline systems on the 2014 SFV dataset using the unofficial key for scoring. We note that our *Union* does not produce a recall of 1.0 on the unofficial scorer due to our single-valued slot selection strategy for multiple systems. As discussed earlier for the single-valued slot, we include the slot fill with highest confidence (which may not necessarily be correct) and thus may not match the unofficial scorer.

Table 8 gives the performance of all our supervised approaches along with the baselines on the common systems dataset using the unofficial key for scoring. UIUC is one of the two teams participating in the SFV 2014 task and the only team to report results, but they report 6 different system configurations and we show their best performance.

5.2 Discussion

Our results indicate that stacking with provenance information and relation type gives the best performance using both the official ESF scorer as well as the unofficial scorer that excludes the human-generated slot fills. Our stacking approach that uses the 10 systems common between 2013 and 2014 also outperforms the ensembling baselines that have the advantage of using *all 65* of the 2014 systems. Our stacking approach would presumably perform even better if we had access to 2013 training data for all 2014 systems.

Of course, the best-performing ESF system for 2014 did not have access to the pooled slot fills of all participating systems. Although pooling the results has an advantage, naive pooling methods such as the ensembling baselines, in particular the voting approach, do not perform as well as our stacked ensembles. Our best approach outperforms the best baseline for both the datasets by at least 6 F1 points using both the official and unof-

ficial scorer.

As expected the *Union* baseline has the highest recall. Among the supervised approaches, stacking with document provenance produces the highest precision and is significantly higher (approximately 5%) than the approach that produces the second highest precision. As discussed earlier, we also scored our approaches on the unofficial scorer so that we can compare our results to the UIUC system that was the best performer in the 2014 SFV task. Our best approach beats their best system configuration by a F1 score of 12 points. Our stacking approach also outperforms them on precision and recall by a large margin.

6 Related Work

Our system is part of a body of work on increasing the performance of relation extraction through ensemble methods.

The use of *stacked generalization* for information extraction has been demonstrated to outperform both majority voting and weighted voting methods (Sigletos et al., 2005). In relation extraction, a stacked classifier effectively combines a supervised, closed-domain Conditional Random Field-based relation extractor with an open-domain CRF Open IE system, yielding a 10% increase in precision without harming recall (Banko et al., 2008). To our knowledge, we are the first to apply stacking to KBP and the first to use provenance as a feature in a stacking approach.

Many KBP SFV systems cast validation as a single-document problem and apply a variety of techniques, such as rule-based consistency checks (Angeli et al., 2013), and techniques from the well-known Recognizing Textual Entailment (RTE) task (Cheng et al., 2013; Sammons et al., 2014). In contrast, the 2013 *JHUAPL* system aggregates the results of many different extractors using a constraint optimization framework, exploiting confidence values reported by each input system (Wang et al., 2013). A second approach in the *UI_CCG* system (Sammons et al., 2014) aggregates results of multiple systems by using majority voting.

In the database, web-search, and data-mining communities, a line of research into "truth-finding" or "truth-discovery" methods addresses the related problem of combining evidence for facts from multiple sources, each with a latent credibility (Yin et al., 2008). The *RPI_BLENDER* KBP system (Yu et al., 2014) casts SFV in this framework, using a graph propagation method that modeled the credibility of systems, sources, and response values. However they only report scores on the 2013 SFV data which contain less complicated and easier queries compared to the 2014 data. Therefore, we cannot directly compare our system's performance to theirs.

Google's Knowledge Vault system (Dong et al., 2014) combines the output of four diverse extraction methods by building a boosted decision stump classifier (Reyzin and Schapire, 2006). For each proposed fact, the classifier considers both the confidence value of each extractor and the number of responsive documents found by the extractor. A separate classifier is trained for each predicate, and Platt Scaling (Platt, 1999) is used to calibrate confidence scores.

7 Conclusion

This paper has presented experimental results showing that *stacking* is a very promising approach to ensembling KBP systems. From our literature survey, we observe that we are the first to employ stacking and combine it with provenance information to ensemble KBP systems. Our stacked meta-classifier provides an F1 score of 50.1% on 2014 KBP ESF, outperforming the best ESF and SFV systems from the 2014 competition, and thereby achieving a new state-of-the-art for this task. We found that provenance features increased accuracy, highlighting the importance of provenance information (even without accessing the source corpus) in addition to confidence scores for ensembling information extraction systems.

8 Acknowledgements

We thank the anonymous reviewers for their valuable feedback. This research was supported by the DARPA DEFT program under AFRL grant FA8750-13-2-0026.

References

Gabor Angeli, Arun Chaganty, Angel Chang, Kevin Reschke, Julie Tibshirani, Jean Y Wu, Osbert Bastani, Keith Siilats, and Christopher D Manning. 2013. Stanford's 2013 KBP system. In *Proceedings of the Sixth Text Analysis Conference (TAC2013)*.

Michele Banko, Oren Etzioni, and Turing Center. 2008. The tradeoffs between open and traditional

relation extraction. In *ACL08*, volume 8, pages 28–36.

Xiao Cheng, Bingling Chen, Rajhans Samdani, Kai-Wei Chang, Zhiye Fei, Mark Sammons, John Wieting, Subhro Roy, Chizheng Wang, and Dan Roth. 2013. Illinois cognitive computation group UI-CCG TAC 2013 entity linking and slot filler validation systems. In *Proceedings of the Sixth Text Analysis Conference (TAC2013)*.

T. Dietterich. 2000. Ensemble methods in machine learning. In J. Kittler and F. Roli, editors, *First International Workshop on Multiple Classifier Systems, Lecture Notes in Computer Science*, pages 1–15. Springer-Verlag.

Pedro Domingos, Stanley Kok, Daniel Lowd, Hoifung Poon, Matthew Richardson, and Parag Singla. 2008. Markov logic. In Luc De Raedt, Paolo Frasconi, Kristian Kersting, and Stephen Muggleton, editors, *Probabilistic Inductive Logic Programming*, volume 4911 of *Lecture Notes in Computer Science*, pages 92–117. Springer.

Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 601–610. ACM.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. 2009. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18.

John C. Henderson and Eric Brill. 1999. Exploiting diversity in natural language processing: Combining parsers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-99)*, pages 187–194, College Park, MD.

Yu Hong, Xiaobin Wang, Yadong Chen, Jian Wang, Tongtao Zhang, Jin Zheng, Dian Yu, Qi Li, Boliang Zhang, Han Wang, et al. 2014. RPI BLENDER TAC-KBP2014 knowledge base population system. *Proceedings of the Seventh Text Analysis Conference (TAC2014)*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Ted Pedersen. 2000. A simple approach to building ensembles of naive Bayesian classifiers for word sense disambiguation. In *Proceedings of the Meeting of the North American Association for Computational Linguistics*, pages 63–69.

John C. Platt. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In Peter J. Bartlett, Bernhard Schölkopf, Dale Schuurmans, and Alex J. Smola, editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, Boston.

Lev Reyzin and Robert E Schapire. 2006. How boosting the margin can also boost classifier complexity. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 753–760. ACM.

Benjamin Roth and Dietrich Klakow. 2010. Cross-language retrieval using link-based language models. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 773–774. ACM.

Benjamin Roth, Tassilo Barth, Michael Wiegand, et al. 2013. Effective slot filling based on shallow distant supervision methods. *Proceedings of the Seventh Text Analysis Conference (TAC2013)*.

Mark Sammons, Yangqiu Song, Ruichen Wang, Gourab Kundu, et al. 2014. Overview of UI-CCG systems for event argument extraction, entity discovery and linking, and slot filler validation. *Proceedings of the Seventh Text Analysis Conference (TAC2014)*.

Georgios Sigletos, Georgios Paliouras, Constantine D Spyropoulos, and Michalis Hatzopoulos. 2005. Combining information extraction systems using voting and stacked generalization. *The Journal of Machine Learning Research*, 6:1751–1782.

Joseph Sill, Gábor Takács, Lester Mackey, and David Lin. 2009. Feature-weighted linear stacking. *arXiv preprint arXiv:0911.0460*.

Sameer Singh, Limin Yao, David Belanger, Ariel Kobren, Sam Anzaroot, Michael Wick, Alexandre Passos, Harshal Pandya, Jinho Choi, Brian Martin, and Andrew McCallum. 2013. Universal schema for slot filling and cold start: UMass IESL.

Mihai Surdeanu and Heng Ji. 2014. Overview of the English slot filling track at the TAC2014 Knowledge Base Population Evaluation. In *Proceedings of the Seventh Text Analysis Conference (TAC2014)*.

Mihai Surdeanu. 2013. Overview of the TAC2013 knowledge base population evaluation: English slot filling and temporal slot filling. In *Proceedings of the Sixth Text Analysis Conference (TAC 2013)*.

I-Jeng Wang, Edwina Liu, Cash Costello, and Christine Piatko. 2013. JHUAPL TAC-KBP2013 slot filler validation system. In *Proceedings of the Sixth Text Analysis Conference (TAC2013)*.

Anurag Wazalwar, Tushar Khot, Ce Zhang, Chris Re, Jude Shavlik, and Sriraam Natarajan. 2014. TAC KBP 2014 : English slot filling track DeepDive with expert advice. In *Proceedings of the Seventh Text Analysis Conference (TAC2014)*.

Matthew Whitehead and Larry Yaeger. 2010. Sentiment mining using ensemble classification models. In Tarek Sobh, editor, *Innovations and Advances in Computer Sciences and Engineering*. Springer Verlag, Berlin.

David H. Wolpert. 1992. Stacked generalization. *Neural Networks*, 5:241–259.

Xiaoxin Yin, Jiawei Han, and Philip S Yu. 2008. Truth discovery with multiple conflicting information providers on the web. *Knowledge and Data Engineering, IEEE Transactions on*, 20(6):796–808.

Dian Yu, Hongzhao Huang, Taylor Cassidy, Heng Ji, Chi Wang, Shi Zhi, Jiawei Han, Clare Voss, and Malik Magdon-Ismail. 2014. The wisdom of minority: Unsupervised slot filling validation based on multi-dimensional truth-finding. In *Proc. The 25th International Conference on Computational Linguistics (COLING2014)*.