Consensus Maximization Fusion of Probabilistic Information Extractors

Miguel Rodríguez and **Sean Goldberg** and **Daisy Zhe Wang**
University of Florida, Dept of Computer Science, Gainesville, FL, USA
{mer,sean,daisyw}@cise.ufl.edu

Abstract

Current approaches to Information Extraction (IE) are capable of extracting large amounts of facts with associated probabilities. Because no current IE system is perfect, complementary and conflicting facts are obtained when different systems are run over the same data. Knowledge Fusion (KF) is the problem of aggregating facts from different extractors. Existing methods approach KF using supervised learning or deep linguistic knowledge, which either lack sufficient data or are not robust enough. We propose a semi-supervised application of Consensus Maximization to the KF problem, using a combination of supervised and unsupervised models. Consensus Maximization Fusion (CM Fusion) is able to promote high quality facts and eliminate incorrect ones. We demonstrate the effectiveness of our system on the NIST Slot Filler Validation contest, which seeks to evaluate and aggregate multiple independent information extractors. Our system achieved the highest F1 score relative to other system submissions.

1 Introduction

The abundance of unstructured text on the web such as news, discussion forums, wiki pages, etc. has increased the interest of the research community in the automatic extraction of information at scale. Information extractors can be used to construct or expand Knowledge Bases (KBs) through a process known as Knowledge Base Population (KBP) or Construction. Facts in a KB are typically modeled as (`subject`, `relation`, `object`) triples such as (*Facebook*, *org:city_of_headquarters*, *Menlo Park*).

No current information extractor is perfectly accurate and different models exhibit different strengths and weaknesses. As a result, many state-of-the-art KBs in academia and industry employ multiple complementary information extractors for KBP. NELL(Mitchell and Fredkin, 2014) employs rules, statistically-learned pattern extractors, and context extractors among others. YAGO (Suchanek et al., 2007) uses heuristic extractors at text and ontological levels and Google has multiple extractors crawling text, tables, and HTML.

Different extraction systems may also agree or disagree on the information they extract. Consider three systems that extract the facts (*Facebook*, *org:city_of_headquarters*, *Menlo Park*), (*Facebook*, *org:city_of_headquarters*, *Palo Alto*), and (*Facebook*, *org:city_of_headquarters*, *Menlo Park*) with probabilities 0.6, 0.3, and 0.5 respectively. The *Palo Alto* extraction is erroneous and should be removed. The two *Menlo Park* extractions should be promoted by agreement and have their confidences increased.

The aggregation of facts from multiple extractors into a single probabilistic triple is known as Knowledge Fusion (KF) (Dong et al., 2014) and can be modeled as an ensemble learning problem. Previous ensemble approaches at the output layer divide into unsupervised and supervised methods (Gao et al., 2010). Unsupervised methods establish a consensus or *majority vote* among extractors without distinguishing the merit of each, but perform poorly if all the extractors are weak. Supervised methods such as *stacking* achieve better performance by learning weights for each extractor and combining them as a weighted sum. The difficulty in obtaining training

Proceedings of NAACL-HLT 2016, pages 1208–1216,
San Diego, California, June 12-17, 2016.

data usually results in high precision, but low recall among all facts.

As a solution to the low recall problem we present a probabilistic ensemble fusion model based on Consensus Maximization (CM) (Gao et al., 2009), which is a semi-supervised learning method able to combine the strengths of both supervised and unsupervised approaches. In the knowledge fusion domain, where the number of unsupervised systems can be rather large compared to those with manually labeled data, Consensus Maximization Fusion is able to leverage both for improved performance.

We apply our CM Fusion approach to the NIST Slot Filling Validation (SFV) task, an ensemble learning problem that aims to combine multiple information extractors participating in the NIST English Slot Filling (ESF) task. Our experiments show an improved F1 score relative to the current state-of-the-art SFV systems.

We make the following overall contributions in this paper:

- Present a novel probabilistic fusion system that incorporates Consensus Maximization to solve the Knowledge Fusion problem.
- Develop an application of our system to the NIST Slot Filling Validation task.
- Outline an evaluation of our system that improves upon the previous state-of-the-art F1 score by 28%.

Though we choose to focus on the SFV task in this paper, there are clear applications beyond this and to the Knowledge Fusion problem in general. The remainder of this paper is organized as follows. In Section 2, we discuss background material on the ESF and SFV tasks as well as the Consensus Maximization algorithm. Section 3 outlines our CM Fusion system and how it maps into the knowledge fusion problem. Our experiments are detailed in Section 4 and we conclude in Section 5.

2 Background

Here we present the appropriate background knowledge on the English Slot Filling (ESF) and Slot Filling Validation (SFV) tasks that are part of the NIST Text Analysis Conference (TAC). We also introduce the Consensus Maximization framework that forms the foundation of CM Fusion.

2.1 Knowledge Base Construction: Slot Filling

A knowledge base is a repository of information about people, places, and things. The usual representation is as (`subject`, `relation`, `object`) triple. The subject and object are entities such as *Facebook* or *Menlo Park*. The relation is some property that holds between the subject and object and usually adheres to a fixed ontology such as *headquarters_in*. Knowledge Base population involves the generation of triples from unstructured text sources.

To facilitate and encourage further research into KBP, NIST has organized a series of workshops known as the Text Analysis Conference (TAC). The English Slot Filler (ESF) task involves connecting a (`subject`, `relation`, $\star$) pair with a set of corresponding `object` attributes. Teams compete with each other to develop the best system for the job (Surdeanu and Ji, 2014).

Each team receives as input a set of queries in XML format and a text corpus. The queries are empty slots such (*Facebook*, *org:city_of_headquarters*, *) and the corpus is a formatted set of web pages, newswire, and discussion forums. For each query slot, the systems extract the appropriate attribute as either a single value or list of values or `NIL`. Overall evaluation is determined by final F1 score across all queries. Query evaluation occurs after submission by a team of human judges. Teams do not know their final accuracy at submission time.

2.2 Ensembling ESF: Slot Filling Validation

The massive quantity of data makes manual labeling impossible and thus system evaluation very difficult. A parallel TAC task is Slot Filling Validation (SFV), which aims to develop a meta-classifier for the purpose of validating individual systems. SFV systems receive as input the set of query results from each ESF system. Without any truth information, they must evaluate the evidence from each query and return a final slot value. The process of sorting conflicting and complementary information from different systems and aggregating into a unified KB is equivalent to the Knowledge Fusion problem.

Sys.	Slot Filler	Provenance	Prob
03_1	Menlo Park	D1:683-692	0.087
03_2	Menlo Park	D1:683-692	0.197
12_4	Menlo Park	D2:655-665	0.987
10_3	Menlo Park	D3:683-692	1.000
13_3	San Francisco	D4:974-986	1.000
07_4	San Francisco	D5:3534-3544	0.210
09_1	Chinatown	D6:3520-3529	0.200
16_1	California	D6:7250-7258	1.000
10_2	California	D7:263-267	1.000

Table 1: Slot fillers extracted by multiple systems for the query (*Facebook*, *org:city_of_headquarters*). The columns represent each system, their response, document provenance, and extracted probabilities.

Table 1 shows an example set of query results from different systems for the (*Facebook*, *org:city_of_headquarters*, *) slot. Included with the slot filler are provenance information about where the filler was mentioned in the corpus and the system's confidence probability.

As stated in the introduction, previous work in SFV has used majority voting (Sammons et al., 2014) or stacking (Viswanathan et al., 2015) to combine the output of multiple ESF systems. The lack of ground truth motivated the majority voting approach of (Sammons et al., 2014), which performs decently in precision and recall. Because of the annual nature of the TAC-KBP competition, some systems repeat submissions in successive years. While current truth data is not available, for a small number of systems there is data available from previous years on a different set of queries. (Viswanathan et al., 2015) uses those systems as training data in a stacking ensemble. Viewing each triple as a binary classification problem into true or false, they extract features based on the probabilities each ESF system gave to each fact (or 0.0 if they did not extract the fact) and train an L1-regularized SVM with a linear kernel. While this gives high precision, the lack of systems participating in multiple years leads to very low recall among all extracted facts.

2.3 Consensus Maximization

Consensus Maximization (Gao et al., 2009) is an ensemble learning method that merges both supervised and unsupervised learning models into a single cohesive framework. While unsupervised models don't provide class labels, they do provide constraints that reduce the hypothesis space of the overall learning problem. Examples in the same cluster are likely to receive the same class label and may improve prediction accuracy through increased model diversity.

Like majority voting and stacking, CM operates at the output level of each individual model and aims to predict the conditional probabilities of each example belonging to each class. By providing class labels, supervised models naturally partition the example space into groups and give those groups labels. Unsupervised models do the same partitioning, but cannot label the groups. Using the supervised methods and their examples as a starting point, CM propagates labels to other unsupervised clusters containing the same examples and to other examples within those clusters.

Formally, the framework of Consensus Maximization is a constrained optimization problem over a bipartite graph. The two sets of nodes are object nodes and group nodes. Each Object node represents a single example given to each system for classification/clustering. Group nodes represent within-system partitions. There are as many partitions as there are classes. For c classes and m systems, there will be $v = cm$ total groups.

The conditional probabilities are expressed in the form of matrices $U_{n\times c}$ for objects and $Q_{v\times c}$ for groups, where rows are object or group nodes from the graph and columns are classes. Rows or columns of the matrix are denoted with the vector $\vec{u_i}$ or $\vec{q_i}$. Each element $u_{iz} \in U_{n\times c}$ represents the conditional probability of object i belonging to class z. Similarly, each element $q_{jz} \in Q_{n\times v}$ represents the conditional probability of group j belong class z as shown in the following equations:

$$u_{iz} = P(y = z|x_i) \tag{1}$$

$$q_{jz} = P(y = z|g_i) \tag{2}$$

The adjacency matrix of the bipartite graph is represented by $A_{n\times v}$ where $a_{ij} = 1$ when x_i has been assigned to group g_j by its respective model, or 0 otherwise. Initial group label assignments are provided by supervised models and encoded in the ma-

trix $Y_{v\times c}$, where $y_{jz} = 1$ if the group g_j comes from supervised learning and belongs to class z, or 0 otherwise. The sum of the rows of Y determines if a group node belongs to a supervised or unsupervised model. Let $k_j = \sum_{z=1}^{c} y_{jz}$ binarily represent such scenario. Note that $k_j = 0$ when there is no supervised evidence for group j. The consensus among models is formulated as the following optimization problem:

$$\begin{aligned}\min_{Q,U}(&\sum_{i=1}^{n}\sum_{j=1}^{v} a_{ij} \parallel \vec{u_i} - \vec{q_j} \parallel^2 \\ &+ \alpha \sum_{j=1}^{v} k_j \parallel \vec{q_j} - \vec{y_j} \parallel^2 \\ &+ \beta \sum_{i=1}^{n} h_i \parallel \vec{u_i} - \vec{f_i} \parallel^2) \\ s.t& \\ \vec{u_i} \geq \vec{0}, &\mid \vec{u_i} \mid = 1, i = 1, \ldots, n \\ \vec{q_j} \geq \vec{0}, &\mid \vec{q_j} \mid = 1, j = 1, \ldots, v\end{aligned} \tag{3}$$

The first term finds consensus across models by minimizing the deviation between the conditional probability of object i in the vector $\vec{u_i}$, and the conditional probability of the groups, $\vec{u_j}$, where it has been originally assigned by the input models. The second term penalizes deviations of the group probability $\vec{q_j}$ from its initial assignment $\vec{y_j}$, with α being the penalization factor that has to be paid for violating supervised predictions. Consensus Maximization does not require labeled objects to estimate the probability matrices Q and U. Nevertheless, a small amount known labels can help bias the optimization and improve the model. If l objects have known labels, the matrix $F_{n\times c}$ encodes this information such that $f_{iz} = 1$ when x_i's true label is z and 0 otherwise. In this matrix, i is the index of an object node, and z a class label. The sum of rows in F determine if an object has a label assigned, $h_i = \sum_{z=1}^{c} f_{iz}$. For instance, $\vec{f_i} = \vec{0}$ represents an objects without labeled data. The third term in the objective function penalizes deviations of the conditional probability of an object $\vec{u_i}$ from its known label $\vec{f_i}$; when $h_i = 0$ this term is dismissed. Labeled variables incorporated in CM are not final. The value of β is the penalization factor for violating label constraints. For a fuller treatment of the methodology, see (Gao et al., 2009).

3 Consensus Maximization Fusion

In this section we describe the ensemble CM Fusion system we built for combining multiple information extractors. Similar to (Viswanathan et al., 2015), we view the slot filling problem in terms of a binary classification task. The set of all extractions across all systems produce an initial knowledge base of facts. This KB is noisy and redundant and contains a lot of conflicting and complementary evidence. For each fact in the KB, we fuse decisions coming from each system into a true class and a false class.

Figure 1 shows the architecture of the system. Offline all individual extractors operate over the same corpus and produce their extractions. A preprocessing step canonicalizes strings and clusters systems producing the same extraction. The feature extraction step converts set of probabilities about each system's fact into the unsupervised feature vector. For the systems for which previous years evaluation data is available we train 6 different meta-classifiers. Their decisions on the same corpus comprise the supervised feature vector. The supervised and unsupervised data are passed into the consensus maximization component that produces a final aggregated probability for each value. As part of a final cleaning process, constraints are applied that remove certain mutually exclusive conflicting facts.

For the remainder of this section we describe each component of our Consensus Maximization Fusion system in more detail.

3.1 Preprocessing

Our system takes as input extractions produced using a number of information extractors that differ in their methodology and application. Their output is uniform and their "filled slots" correspond to a set of extracted facts. Each fact is processed by transforming all text into lower case and deleting trailing spaces. Using exact string match we map each fact onto a cluster of the multiple systems that extracted it.

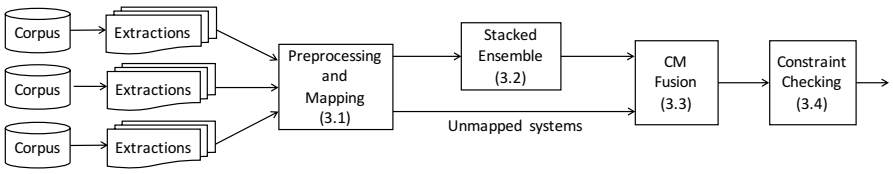

Figure 1: Consensus Maximization Fusion system components and pipeline.

3.2 Stacked Ensemble

Consensus Maximization Fusion is able to take into account supervised and unsupervised systems for knowledge fusion. The supervised portion is able to weight certain systems using previously labeled data and combine their decisions into a meta-classifier. The unsupervised portion operating over unlabeled data treats every system equally and is closer in spirit to a majority vote.

Of the extractors submitting results to ESF, about 10% had participated in previous years where labeled data was available. This idea was used in (Viswanathan et al., 2015) to generate a series meta-classifiers using stacking. A meta-classifier combines the outputs of multiple systems using learned weights. (Viswanathan et al., 2015) generate a total of 6 meta-classifiers for comparison that differ in classifier type and feature vector. The basic feature vector includes one entry for each system and with the value being that system's extraction probability. Two additional feature vectors were generated by adding relational information for each system and both relational information and provenance information. These three feature vectors were independently trained using logistic regression (LR) and SVM for a total of 6 meta-classifiers. CM Fusion runs each trained meta-classifier over the same ESF corpus and uses their output as an additional 6 systems in the ensemble combination. Though we use only 6 our method generalizes to any number of meta-classifiers.

3.3 Fusing Systems

In total there are S systems submitting runs to be ensemble by CM Fusion. N of these systems have evaluation data for the training a total of M meta-classifiers. The top half of Figure 2 shows the collection of supervised meta-classifiers and remaining unsupervised systems.

Symbol	Definition
$o_1, ..., o_k$	Unique facts extracted
$g_1, ..., g_{2m}$	Group nodes from meta-classifiers
$g_{2m+1}, ..., g_t$	Group nodes from unmapped runs
$A = [a_{ij}]$	Indicator of fact i in group j
$U = \vec{u_i}$	Conditional prob of $o_i = true$
$Q = \vec{q_i}$	Indicator of unmapped run $g_i = yes$
$Y = \vec{y_j}$	Indicator of meta-classifier $g_j = yes$
$F = \vec{f_i}$	o_i label if known a priori

Table 2: CM Fusion notations mapped to the SFV task.

Consensus Maximization operates as a bipartite graph between object nodes representing examples to classify and group nodes representing classes within each system. In the knowledge fusion domain, each object node pertains to a specific fact triple under consideration. The two classes for each system (*Yes*/*No*) all combined represent the set of group nodes. The translation component of CM Fusion generates the bipartite graph in Figure 2 as input to the consensus maximization component.

The overall goal of consensus maximization is to combine labeled predictions from the M supervised meta-classifiers and consensus predictions from the unsupervised systems using a constrained optimization framework. As recalled in Section 2, CM optimizes U and Q, which are conditional probabilities between object nodes (facts) and classes (*Yes*/*No*) and between group nodes (output classes for each system) and classes (*Yes*/*No*) respectively. Because we are only interested in the *Yes* class probabilities, the matrices collapse into vectors $\vec{u}$ and $\vec{q}$ as per equation 3. $\vec{u_i}$ is initialized as an uninformed prior with value 0.5 for each object node. Since unsupervised output labels are known a priori, $\vec{q_j}$ remains unchanged in the optimization process and encodes the pertinence of each group node j to a *Yes*/*No* class. Finally $\vec{y_j}$ includes the supervised meta-classifier output for each group. Running the

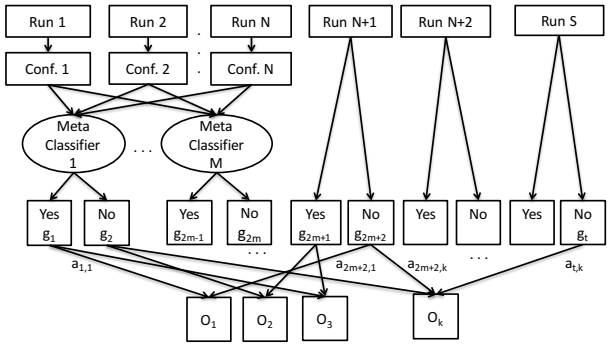

Figure 2: CM Bipartite graph for SFV. Object nodes correspond to facts. Each of S systems partition facts into *Yes* or *No* classes, which act as groups in CM translation.

optimization program generates a final $\vec{u_i}$ that gives a posterior probability for each fact. Table 2 maps all CM symbols in equation 3 to their corresponding elements in the SFV problem definition.

3.4 Enforced Constraints

After applying consensus maximization, there may be some facts that can be eliminated using functionality constraints. A slot is functional when it has only one possible value for its filler. Functional slots with different fillers may have certain facts eliminated based on mutual exclusion. As an example, consider the various slot filler responses in Table 1. An organization can only have its headquarters in one city at any given time. In this case *Menlo Park* should be chosen based on it having the maximum probability and all other facts eliminated. When the extraction probability is the same for multiple systems, one of the extractions is chosen at random. Nevertheless, this is hardly ever the case.

4 Experiments

We evaluate our system using a collection of queries supplied by TAC-KBP SFV. In this section we more fully describe the dataset and our experiment methodology. We compare to the current state-of-the-art ESF and SFV systems and show an improvement in F1 score. Finally, we provide an analysis of our results.

4.1 Datasets

Three datasets were used for training and testing spread across the three years that the English Slot Filling task has been run. Each team in the competition submits multiple models vying for the highest score on unseen training data. Each submission is viewed as a different system in our ensemble. 2013 contains 52 systems. 2014 contains 65 and 2015 contains 69. In 2013 and 2014 we use only ESF data, but 2015 adds Cold Start Knowledge Base (CSKB) data. The total number of labeled queries in both 2013 and 2014 were 100 each. 2015 had 9340 unlabeled queries. In addition, 2015 had 164 labeled queries supplied for initial system assessment that was incorporated into the training data for 2015 submissions.

We performed two distinct experiments and compared multiple baselines for each. Table 3 shows results where we trained on 2013 only and tested on 2014 data only. Table 4 results are for training on 2013 and 2014 data and testing on 2015 data.

4.2 0-Hop and 1-Hop Queries

The 2015 ESF task introduced more complex queries into the fold. 1-hop queries use the result of a previous query to answer a new query. For example, consider a query asking for all companies headquartered in the same city as Facebook. The equivalent 1-hop query is (*Facebook*, *org:city_of_headquarters*, *?x*), (*?x*, *gpe:headquarters_in_city*, ***). The second part of the query can be seen as a join to the first part using a common answer in both. The terminology exploits the idea of "hopping" from one query to another. By contrast, a regular slot filling task is a 0-hop query.

4.3 Evaluation

All systems were evaluated using the standard metrics of precision, recall, and F1 which is the harmonic mean of precision and recall. Precision is the amount of correctly extracted facts compared to the total facts extracted by the system. Recall is the amount of correct facts compared all facts in the ground truth. In review:

$$Recall(R) = \frac{\text{Correct}}{\text{Total}} \tag{4}$$

$$Precision(P) = \frac{\text{Correct}}{\text{Extracted}} \tag{5}$$

$$F1 = 2\frac{PR}{P+R} \tag{6}$$

Method	P	R	F1
LR	0.648	0.335	0.441
LR + REL	0.662	0.343	0.452
LR + PROV + REL	0.634	0.374	0.470
SVM	0.639	0.319	0.425
SVM + REL	0.720	0.299	0.422
SVM + PROV + REL	**0.729**	0.298	0.423
MV	0.467	0.447	0.457
Stanford	0.585	0.298	0.395
CM Fusion	0.549	**0.538**	**0.544**

Table 3: Result comparisons for the testing on the 2014 SFV task with training done on 2013 ESF data. The first 6 methods correspond to stacking while the others correspond to majority vote, the best performing 2014 ESF system, and CM Fusion. By increasing recall, CM Fusion has the highest F1 among all baselines.

	P	R	F1	Queries
2013 & 2014	**0.528**	0.481	0.504	0-Hop
2014 only	0.477	**0.539**	**0.506**	
BBN	0.493	0.391	0.436	
2013 & 2014	**0.393**	0.097	0.155	1-Hop
2014 only	0.314	0.141	0.194	
Stanford	0.184	**0.304**	**0.229**	
2013 & 2014	**0.503**	0.307	0.381	ALL
2014 only	0.436	**0.358**	**0.393**	
BBN	0.378	0.261	0.309	

Table 4: Summary of submissions to the SFV 2015 task for different query types. We trained the meta-classifiers on 2014 ESF data or 2013 and 2014 ESF data. Comparison is made to the highest scoring individual ESF system by F1.

The 0-hop queries are scored trivially on the correctness of the slot filler. The 1-hop queries require two verifications to undergo scoring. The 0-hop query from which it was derived must both exist and be correct. Any slot fillers that don't meet this criteria are omitted even if their 1-hop slot was ultimately correct. For example, the 1-hop extraction (*Palo Alto*, *gpe:headquarters_in_city*, *Hewlett-Packard*) will be ignored even though it is correct because the 0-hop query (*Facebook*, *gpe:city_of_headquarters*, *Palo Alto*) that derives it is incorrect (Facebook is headquarted in Menlo Park).

4.4 Results

Table 3 shows results using 2013 ESF systems as training data for the meta-classifiers and 2014 ESF systems as testing data. The first 6 rows are a combination of classifiers and feature vectors using stacking as described in (Viswanathan et al., 2015). MV refers to the majority voting described (Sammons et al., 2014). Majority voting accepts a fact if a certain number of systems above some learned threshold have extracted it. Both of these systems showcased results on the 2014 SFV task. The Stanford system (Angeli et al., 2014) was the best performing ESF system during the 2014 competition. The final row is our CM Fusion algorithm. Only 0-hop queries are used for these results.

Table 4 showcases two of the three runs we officially submitted as part of the recent 2015 SFV task for different query types. For each query, the first two rows refer to our CM Fusion system with different training data and the final row the best performing 2015 ESF system for that particular query type. 2015 results are not officially published yet for either ESF or SFV. Nevertheless, at 2015 workshop announcing the results, CM Fusion was awarded as the top ranked SFV system by F1 score.

4.5 Analysis

The main drawback between previous ensemble systems that utilize only supervised systems is high precision, but very low recall. This is evidenced in the disproportion between precision and recall in the first 6 systems of Table 3. Majority voting, while learning a threshold using supervision, does include some unsupervised consensus.

The main idea behind CM Fusion is to take into account the answers from potentially well-ranked extractors that stacking meta-classifiers omit due lack of training data. CM Fusion outperforms both approaches in terms of F1 by greatly increasing the recall while maintaining high precision. It also produces better results than the best performing 2014 ESF system. On 2015 data in Table 4, both systems outperform the best performing 2015 ESF system.

The benefit of using CM Fusion over other supervised ensemble models is the ability to use unsupervised systems that lack sufficient training data. When these systems agree on extractions, their consensus can be a source of discriminative power which CM Fusion is able to harness. For our submis-

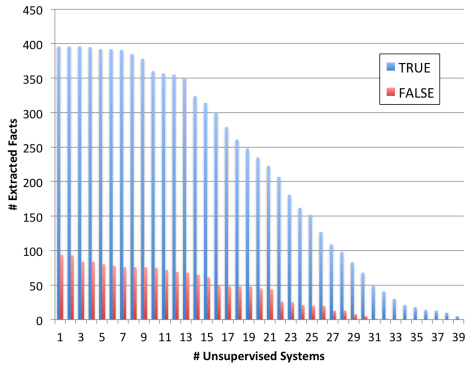

Figure 3: Extracted facts among unsupervised systems that agree with at least one supervised system using 2015 queries. The x-axis shows the minimum number of systems used for consensus.

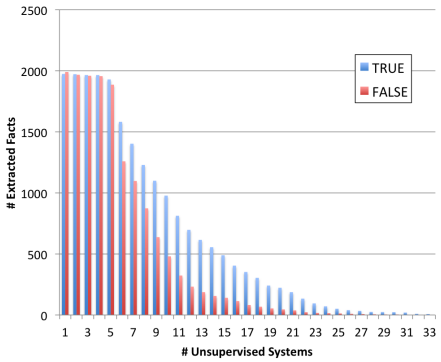

Figure 4: Extracted facts among unsupervised systems using 2015 queries using only unsupervised consensus. The x-axis shows the minimum number of systems used for consensus. The scale is larger than Figure 3 because agreement with a supervised system is not needed.

sion trained only on 2014 ESF data, Figures 3 and 4 break down the difference between extracted facts derivable only from supervised systems and those able to be attained from unsupervised systems.

Specifically, Figure 3 measures the number of extracted facts supported by a consensus of the number of systems on the x-axis that also agree with the output of a supervised system. Blue bars represent the amount of correct extractions and red bars incorrect extractions. When only supervision is used there is good precision across the board. Figure 4 shows the number of extracted facts supported only by a consensus of systems on the x-axis without any supervision. The disparity in the scale of facts extracted supports the recall of supervised systems. When only a few systems agree on an extraction, about as many good facts are extracted as bad ones. As consensus increases, precision greatly improves. This idea of unsupervised consensus is exactly what improves CM Fusion over supervised ensemble approaches.

To understand the impact of the number of unsupervised systems fused and the quality of the fused runs in the ensemble, we applied CM Fusion in an incremental fashion by adding one unsupervised system at a time and scored the produced ensemble at each step. Techniques for ranking unsupervised systems are outside the scope of this work, for simplicity we examine the best- and worst-cases for adding systems incrementally. Ranking each unsupervised system by its final individual F1 score on the SFV 2015 data, we ran CM Fusion adding the best performing run (CMF-BEST) and worst performing run (CMF-WORST) at each step. The results are displayed in Figure 5. For comparison, we also included the results of the best performing ESF system (BBN-F1) and the score of a supervised-only ensemble (SUP-ONLY). Fusing unsupervised systems with lower accuracy negatively affects the quality of the ensemble compared to supervised-only. When ensembleing very similar runs, such as runs submitted by the same team, the diversity of the systems is compromised and may lower the ensemble quality. On the other hand, unsupervised systems with higher accuracy can rapidly increase the quality of the ensemble above the best individual results and reach the highest ensemble score. Nevertheless, more does not necessarily mean better, as shown in the plateau of the best-case plot in Figure 5, when noisy systems are added to the ensemble the F1 score is maintained. The average case would lie somewhere in between the extremes of the best- and worst-case. As is shown, a large number of systems are extremely error-prone, but the combination using CM Fusion produces a result ultimately superior.

5 Conclusions

This paper presented our Consensus Maximization Fusion of multiple probabilistic information extractors. This approach combines supervised stacking

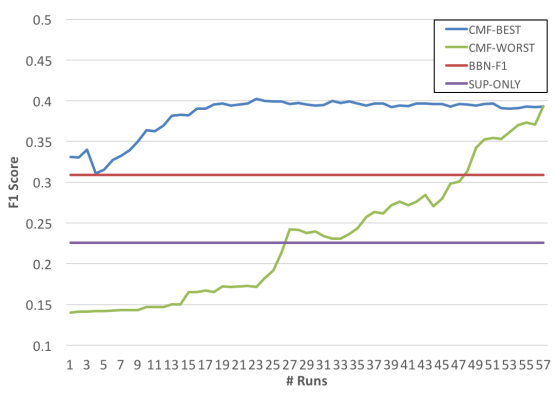

Figure 5: Incremental CM Fusion in terms of adding best-performing (blue) and worst-performing (green) systems one-by-one. Performance is ranked by F1 scores from the 2015 SFV dataset. F1 scores are in the range $[0.03 - 0.309]$ with an average F1 of 0.140

meta-classifiers with unsupervised extraction outputs in an ensemble classifier. Our system outperformed the current state-of-the-art ensemble models submitted to the TAC-KBP Slot Filler Validation task in 2014. CM Fusion was also chosen as the leading system at the 2015 TAC-KBP Workshop. This is the first cross-model ensemble approach that fuses multiple knowledge graphs obtained from both supervised and unsupervised information extractors. Optimal performance is attained when the extractors represent different systems running over the same corpus and the shared extraction density is high.

The canonicalization of facts in CM Fusion represents a new state-of-the-art contribution to entity-centric information extraction compared to traditional document-centric approaches. While our approach has been experimentally verified using TAC-KBP data, it generalizes to overlapping ensemble of knowledge bases. Some such as NELL (Mitchell and Fredkin, 2014) have a small supervised human feedback component and others such as OpenIE (Banko et al., 2007) are entirely unsupervised. Future work concerns using CM Fusion to to align and canonicalize multiple such knowledge bases to solve the knowledge fusion problem.

Acknowledgments

This work is partially supported by NSF IIS Award #1526753, DARPA under FA8750-12-2-0348-2 (DEFT/CUBISM) and graduate fellowships from Fulbright and Sandia National Labs

References

Gabor Angeli, Julie Tibshirani, Jean Y Wu, and Christopher D Manning. 2014. Combining distant and partial supervision for relation extraction. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*.

Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction for the web. In *IJCAI*, volume 7, pages 2670–2676.

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.

Jing Gao, Feng Liang, Wei Fan, Yizhou Sun, and Jiawei Han. 2009. Graph-based consensus maximization among multiple supervised and unsupervised models. In *Advances in Neural Information Processing Systems*, pages 585–593.

Jing Gao, Wei Fan, and Jiawei Han. 2010. On the power of ensemble: Supervised and unsupervised methods reconciled. In *Tutorial on SIAM data mining conference(SDM)*. Citeseer.

Tom Mitchell and E Fredkin. 2014. Never ending language learning. In *Big Data (Big Data), 2014 IEEE International Conference on*, pages 1–1. IEEE.

Mark Sammons, Yangqiu Song, Ruichen Wang, Gourab Kundu, Chen-Tse Tsai, Shyam Upadhyay, Siddarth Ancha, Stephen Mayhew, and Dan Roth. 2014. Overview of ui-ccg systems for event argument extraction, entity discovery and linking, and slot filler validation. *Urbana*, 51:61801.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM.

Mihai Surdeanu and Heng Ji. 2014. Overview of the english slot filling track at the tac2014 knowledge base population evaluation. In *Proc. Text Analysis Conference (TAC2014)*.

Vidhoon Viswanathan, Nazneen Fatema Rajani, Yinon Bentor, and Raymond Mooney. 2015. Stacked ensembles of information extractors for knowledge-base population. In *Proceedings of ACL*.