Stacking With Auxiliary Features: Improved Ensembling for Natural Language and Vision

Nazneen Fatema Rajani
The University of Texas at Austin
Austin, TX 78712
nrajani@cs.utexas.edu

Doctoral Dissertation Proposal

November 7, 2016

Abstract

Ensembling methods are well known in machine learning for improving prediction accuracy. However, they are limited in the sense that they cannot effectively discriminate among underlying component models. Some models perform better at certain types of input instances than other models. The measure of how good a model is can sometimes be gauged from "where" it extracted the output and "why" it made the prediction. This information can be exploited to leverage the component models in an ensemble. In this proposal, we present stacking with auxiliary features that integrates relevant information from multiple sources to improve ensembling. We use two types of auxiliary features – *instance features* and *provenance features*. The instance features enable the stacker to discriminate across input instances while the provenance features enable the stacker to discriminate across component systems. When combined together, our algorithm learns to rely on systems that not just agree on an output but also the provenance of this output in conjunction with the input instance type.

We demonstrate our approach on three very different and difficult problems: Cold Start Slot Filling, Tri-lingual Entity Discovery and Linking, and ImageNet Object Detection. The first two problems are well known tasks in Natural Language Processing, and the third one is in the domain of Computer Vision. Our algorithm obtains state-of-the-art results on the first two tasks and significant improvements on the ImageNet task, thus verifying the power and generality of our approach. We also present a novel approach using stacking for combining systems that do not have training data in an unsupervised ensemble with systems that *do* have training data. Our combined approach achieves state-of-the-art on the Cold Start Slot Filling and Tri-lingual Entity Discovery and Linking tasks, beating our own prior performance on ensembling just the supervised systems.

We propose several short-term and long-term extensions to our work. In the short-term, we focus our work on using more semantic instance-level features for all the three tasks, and use non-lexical features that are language independent for the two NLP tasks. In the long-term we propose to demonstrate our ensembling algorithm on the Visual Question Answering task and use textual/visual explanations as auxiliary features to stacking.

Contents

1 Introduction

Ensembling multiple systems is a well known standard approach to improving accuracy in several machine learning applications (Dietterich, 2000). Ensembles have been applied to a wide variety of problems in all domains of artificial intelligence including Natural Language Processing (NLP) and Computer Vision (CV). In NLP, these have been applied to parsing (Henderson and Brill, 1999), word sense disambiguation (Pedersen, 2000), sentiment analysis (Whitehead and Yaeger, 2010) and information extraction (IE) (Florian et al., 2003; McClosky et al., 2012). In the domain of Computer Vision, these have been used for image classification (Korytkowski et al., 2016), object tracking (Zhou et al., 2014), object detection (Malisiewicz et al., 2011) and zero-shot recognition (Jayaraman and Grauman, 2014). However, these techniques do not learn to adequately discriminate across the component systems and thus are unable to optimally integrate them. Therefore, combining systems intelligently is crucial for improving the overall performance. We seek to integrate knowledge from multiple sources to improve ensembling using a new approach we call Stacking with Auxiliary Features (SWAF) (Rajani and Mooney, 2016a). Stacking (Wolpert, 1992) uses supervised learning to train a meta-classifier to combine multiple system outputs. The auxiliary features enable the stacker to fuse additional relevant knowledge from multiple systems and thus leverage them to improve prediction.

We consider the general machine learning problem of combining outputs from multiple systems to improve accuracy by using auxiliary features. We use two types of auxiliary features – those that enable the stacker to discriminate across instances which we call, the *instance features* and those that enable the stacker to discriminate across component systems which we call, the *provenance features*. Stacking with auxiliary features can be successfully deployed to any problem whose output instances have confidence scores and optionally *provenance* that justifies the output. Provenance indicates the origin of the generated output and can be used to measure reliability of a system output. The idea behind using auxiliary features is that, an output is more reliable, if not just multiple systems produce it but also agree on its provenance as well as there are sufficient supporting instance features. The SWAF algorithm thus requires identifying the instance and provenance features for a given task.

Since stacking uses supervised learning, it requires training data to ensemble multiple systems. However, we would sometimes like to ensemble systems for which we have no historical performance data. For example, due to privacy, some companies may be unwilling to share their performance on arbitrary training sets. Simple methods such as voting permit "unsupervised" ensembling, and several more sophisticated methods have also been developed for this scenario (Wang et al., 2013). However, such methods fail to exploit supervision for those systems for which we *do* have training data. Therefore, we designed an approach that utilizes supervised *and* unsupervised ensembling to exploit the advantages of both (Rajani and Mooney, 2016b). We first use unsupervised ensembling to combine systems without training data, and then use stacking to combine this ensembled system with other systems with available training data. Thus SWAF can handle both systems with and without training data for learning an ensemble.

We use SWAF to demonstrate new state-of-the-art results for ensembling on two of three tasks and significant improvements over the best component system for the third task. All the three tasks are difficult and well-known challenge problems. The first two tasks are in NLP and part of the

NIST Knowledge Base Population (KBP) challenge – *Cold Start Slot-Filling* (CSSF)[1] (Surdeanu and Ji, 2014) and *Entity Discovery and Linking* (EDL) (Ji et al., 2015). The third task is in computer vision and part of the ImageNet 2015 challenge(Russakovsky et al., 2015) – *Object Detection* from images. Auxiliary features for each of the tasks are identified based on the goal and supplementary resources for the problem. We demonstrate results on the 2014 and 2015 iterations of the CSSF task, 2015 and 2016 iterations of the TEDL task and the 2015 iteration of the ImageNet object detection task (Rajani et al., 2015; Rajani and Mooney, 2016a). Our approach using the auxiliary features on each of the three tasks outperforms the individual component systems as well as other ensembling methods such as the Mixtures of Experts model (Jacobs et al., 1991) and *oracle* voting baselines on all three tasks in the 2015 competition; verifying the generality and power of stacking with auxiliary features.

Finally, we propose some short-term and long-term extensions to our completed work. In the short-term, we focus on how our work may be extended to using more semantic instance-level features and to achieve performance on par with English on KBP tasks for the foreign languages such as Chinese and Spanish, by using non-lexical features. Our long-term proposal is a step in the direction of explainable AI, which emphasizes high-preforming models that produce intelligible explanations for their decisions[2]. The idea behind encouraging explainable AI is threefold: (1) Identify reasons for failure when AI is significantly weaker than humans (2) Build trust with users when AI is on par with humans; and (3) Pedagogical explanations when AI is significantly stronger than humans. Our completed work focuses on using provenance that captures the "where?" aspect of the output, on the other hand, our long-term proposal is to use the explanations as auxiliary features so that it adds the "why?" dimension to our existing model. Our proposal on using explanations as features can be divided into two categories (1) using textual explanations as auxiliary features; and (2) using visual explanation as auxiliary features.

[1] `http://www.nist.gov/tac/2015/KBP/ColdStart/guidelines.html`

[2] `http://www.darpa.mil/program/explainable-artificial-intelligence`

2 Background and Related Work

In this section, we review some of the background crucial to our work. First we discuss the Cold Start Slot Filling (CSSF) task which is a type of relation extraction with a fixed ontology. Thereafter, we discuss the Entity Discovery and Linking (EDL) task. We then overview the ImageNet object-detection task and finally discuss ensemble techniques used for these tasks.

2.1 Cold Start Slot Filling (CSSF)

Knowledge Base Population (KBP) is a NLP problem of discovering facts about entities and augmenting them into a Knowledge Base (KB) (Ji and Grishman, 2011). Information Extraction (IE) or Relation Extraction in particular, is a sub-task for KBP. Relation extraction using a fixed ontology is called Slot-Filling (SF). Cold Start Slot Filling (CSSF) is the task of building a KB from scratch just based on query entities and slots. NIST annually conducts the CSSF task (Surdeanu, 2013; Surdeanu and Ji, 2014) in the KBP track of the Text Analysis Conference (TAC).

The goal of CSSF is to collect information (fills) about specific attributes (slots) for a set of entities (queries) from a given corpus. The query entities can be a person (PER), organization (ORG) or geo-political entity (GPE). The slots are fixed and the 2015 task also included the inverse of each slot, for example the slot org:subsidiaries and its inverse org:parents. The evaluation included a total of forty-one slots and their inverses[3]. Some slots (like *per:age*) are *single-valued* while others (like *per:children*) are *list-valued* i.e., they can take multiple slot fillers.

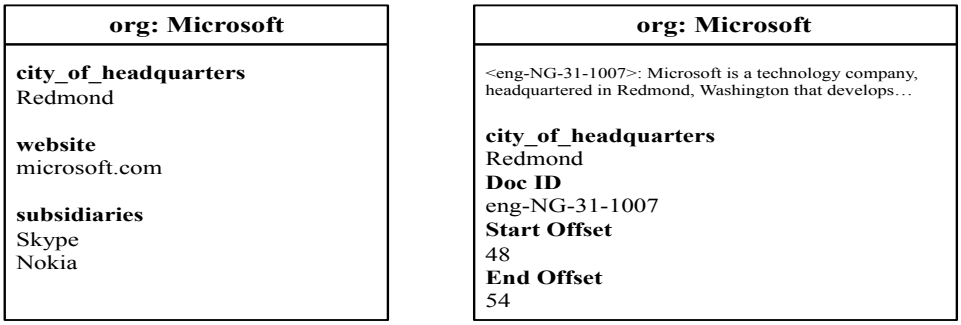


Figure 1: An example query entity and some slot fills for the CSSF task on the left and the expected output format on the right.

The input for CSSF is a set of queries and a text corpus in which to look for information. The queries are provided in an XML format that includes an ID for the query, the name of the entity, and the type of entity (PER, ORG or GPE). The corpus consists of documents from discussion forums, newswire and the Internet, each identified by a unique ID. The output is a set of slot fills for each

[3] https://tac.nist.gov/2015/KBP/ColdStart/guidelines/TAC_KBP_2015_ColdStartTaskDescription_1.1.pdf

query. Along with each slot fill, systems must also provide its *provenance* in the corpus in the form *docid:startoffset-endoffset*, where *docid* specifies a source document and the offsets demarcate the text in this document containing the extracted filler. Systems also provide a confidence score to indicate their certainty in the extracted information. Figure 1 illustrates an example entity and some slot fills as well as the expected output format for the task.

For the CSSF task, participating teams employ a variety of techniques such as distant supervision, universal schema, relevant document extraction, relation-modeling, OpenIE and inference (Finin et al., 2015; Soderland et al., 2015; Kisiel et al., 2015). The top performing 2015 CSSF system (Angeli et al., 2015) leverages both distant supervision (Mintz et al., 2009) and pattern-based relation extraction. Another system, *UMass_IESL* (Roth et al., 2015), uses distant supervision, rule-based extractors, and semi-supervised matrix embedding methods. 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). We are the first to apply stacking to KBP and the first to use provenance as a feature in a stacking approach.

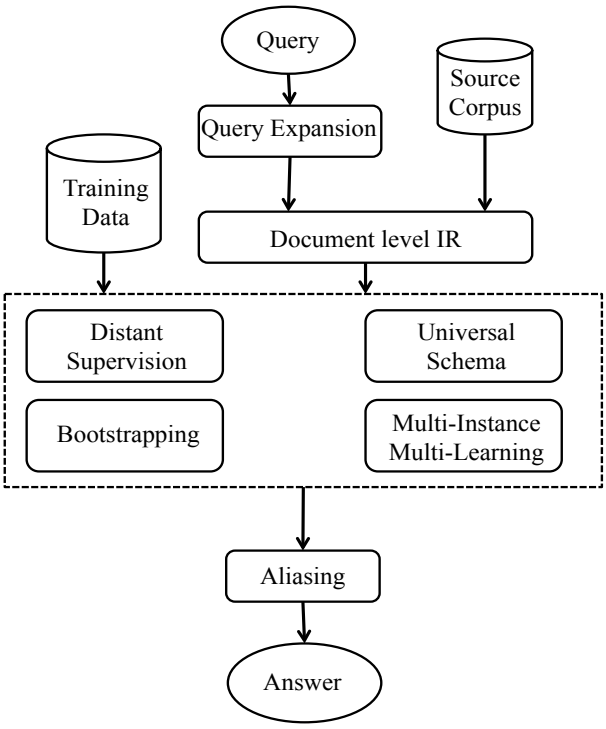


Figure 2: Overview of slot-filling system architecture.

The KBP evaluations also includes the Slot Filler Validation (SFV) task[4] where the goal is to ensemble/filter outputs from multiple slot filling systems. Many KBP SFV systems cast validation

[4] http://www.nist.gov/tac/2015/KBP/SFValidation/index.html

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.

Stacking for information extraction has been demonstrated to outperform both majority voting and weighted voting methods (Sigletos et al., 2005). The FAUST system for biomolecular event extraction uses model combination strategies such as voting and stacking and was placed first in three of the four BioNLP tasks in 2011 (Riedel et al., 2011). 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. 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. Figure 2 gives an overview of a general slot-filling system.

per: Hillary Clinton
Source Corpus Document <doc id="ENG_NW_001429.nw.xml"> : Hillary Clinton Not Talking About '92 Clinton-Gore Confederate Campaign Button.. **FreeBase** <ID= m.0d06m >: Hillary Diane Rodham Clinton is a US Secretary of State, U.S. Senator, and First Lady of the United States. From 2009 to 2013, she was the 67th Secretary of State, serving under President Barack Obama. She previously represented New York in the U.S. Senate. Before that, as the wife of President Bill Clinton, she was First Lady from 1993 to 2001...

Figure 3: Example of an entity mention in a document and its KB entry for the EDL task.

2.2 Entity Discovery and Linking (EDL)

Entity Discovery and Linking (EDL) is another sub-task for KBP. It involves two sub-problems widely known in NLP – (1) Named Entity Recognition (NER); and (2) Disambiguation. NIST annually conducts the EDL task in the KBP track of the Text Analysis Conference (TAC) (Ji et al., 2015). The source corpus provided for the task contains documents in Chinese and Spanish along with English and thus it is called Tr-lingual Entity Discovery and Linking (TEDL).

The goal of TEDL is to discover all entity mentions in a corpus of English, Spanish and Chinese documents. The entities can be a person (PER), organization (ORG), geo-political entity (GPE), facility (FAC), or location (LOC). The FAC and LOC entity types were newly introduced in 2015. The extracted mentions are then linked to an existing English KB entity using its ID. If there is no KB entry for an entity, systems are expected to cluster all the mentions for that entity using a NIL ID.

The input is a corpus of documents in the three languages and an English KB (FreeBase) of entities, each with a name, ID, type, and several relation tuples that allow systems to disambiguate entities. The output is a set of extracted mentions, each with a string, its provenance in the corpus, and a corresponding KB ID if the system could successfully link the mention, or else a mention cluster with a NIL ID. Systems can also provide a confidence score for each mention. Figure 3 illustrates an example entity mention in a document and its corresponding KB entry.

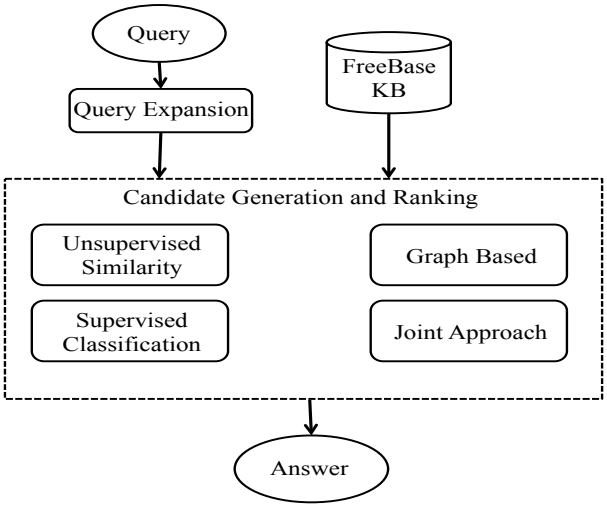


Figure 4: Overview of an entity-linking system architecture.

Participating teams employ a variety of techniques for candidate generation and ranking. The *CMUML* team used a unified graph-based approach to do concept disambiguation and entity linking by leveraging the FreeBase ontology (Fauceglia et al., 2015). The *HITS* team combined local, unsupervised sieves with a global, supervised, joint disambiguation and NIL clustering to build a hybrid system (Heinzerling et al., 2015). The *UI_CCG* focused on the Spanish language sub-task of TEDL, by using Google Translation to translate Spanish documents into English and then using Illinois Wikifier to identify entity mentions and disambiguate them to FreeBase entities (Sammons et al., 2015). The top performing 2015 TEDL system used a combination of deep neural networks and Conditional Random Fields (CRFs) for mention detection and a language-independent probabilistic disambiguation model for entity linking (Sil et al., 2015). A fast and scalable collective entity linking method that relies on stacking was proposed by He et al. (2013). They stack a global predictor on top of a local predictor to collect coherence information from neighboring decisions. Biomedical entity extraction using a stacked ensemble of an SVM and CRF was shown to outperform individual components as well as voting baselines (Ekbal and Saha, 2013). However, there

has been no prior work on ensembling for the TEDL task, and our SWAF approach beats the current state-of-the-art system. Figure 4 gives an overview of a very general entity-linking system.

2.3 ImageNet Object Detection

The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) is a widely known annual competition in Computer Vision and has become the standard benchmark for large-scale object recognition (Russakovsky et al., 2015). The goal of the ImageNet object detection task is to detect all instances of object categories (out of the 200 predefined categories) present in the image and localize them by providing coordinates of the axis-aligned bounding boxes for each instance. The ImageNet dataset is organized according to the WordNet hierarchy and thus the object categories are WordNet *synsets*.

The object detection corpus is divided into training, validation and test sets. The training set consists of approximately 450K images including both positive and negative instances, annotated with bounding boxes; the validation set consists of around 20K images also annotated for all object categories and the test set has 50K images. The output for the task is the image ID, the object category (1-200), a confidence score, and the coordinates of the bounding box. In case of multiple instances in the same image, each instance is mentioned on a separate line. Figure 5 shows a random sample of images with bounding boxes around the detected object categories.

Because of the large scale of the ImageNet dataset, almost all participating teams employ deep learning for image classification. For the object detection in 2015, the top performing team used a deep residual net (He et al., 2015) and several other teams deployed a version of faster R-CNN (Region based Convolutional Neural Networks) with selective search (Ren et al., 2015). Faster R-CNN is a more efficient variant of fast R-CNN (Girshick, 2015) that first uses Region Proposal Networks (RPN) to train an end-to-end network for generating region proposals. There has been some work on stacking for multi-layer object recognition (Peppoloni et al., 2014) but our paper is the first to use stacking for ensembling object *detectors* and we obtain significant improvements over the component systems.

2.4 Ensemble Algorithms

Ensemble methods are models composed of multiple component models that are independently trained and whose predictions are combined in some way to make the overall prediction. These methods vary in the ways the component models are combined and what types of models are combined. Ensemble algorithms are very powerful and thus well known. Some of these popular types of ensembling algorithms are:

- **Boosting** (Freund and Schapire, 1995) refers to building a model from the training data, then creating a second model that attempts to correct the errors from the first model. Models are added until the training set is predicted perfectly or a maximum number of models are added. Boosting focuses on reducing the bias.

- **Bagging** (Breiman, 1996) or Bootstrap Aggregation refers to prediction by generating additional data for training using repetitions to produce multisets of the same cardinality as the original data. Bagging focuses on reducing the variance.

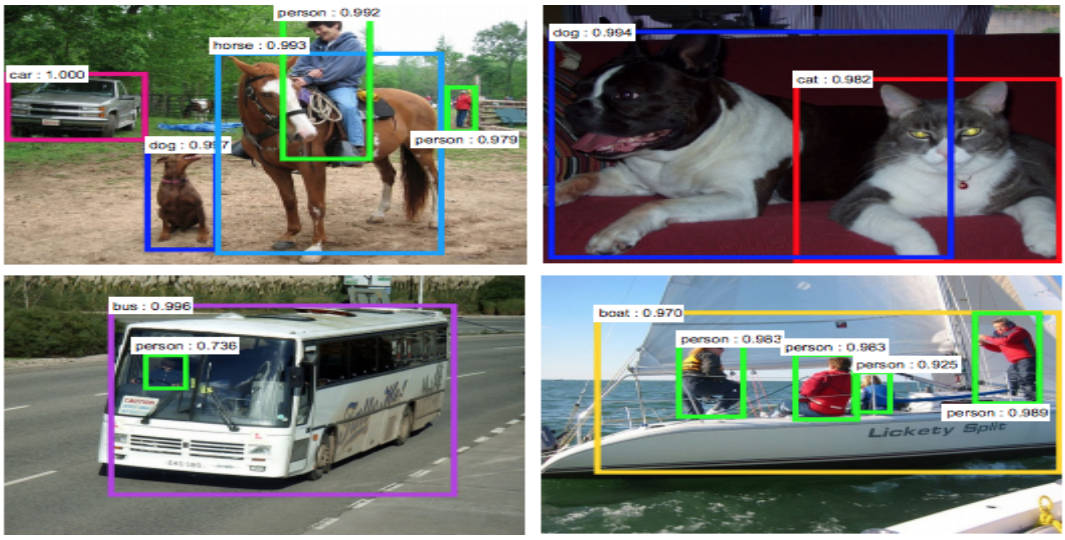


Figure 5: ImageNet Object detection sample images with bounding boxes around categories. Images taken from (Girshick et al., 2014)

- **Stacking** (Wolpert, 1992) refers to building a meta-classifier on top of the component models to estimate the decision based on the outputs produced by the base models.
- **Random Forest** (Liaw and Wiener, 2002) refers to the weighted combination of decision trees trained on random subsets of the data.

In this proposal, we use stacking as our ensembling algorithm and extend it to make it even more powerful by using task specific features which we call as *auxiliary features*.

In the past, many new ensembling algorithms have been proposed. Bipartite Graph Based Consensus Maximization (BGCM) is one such algorithm (Gao et al., 2009). The authors introduce BGCM as a way of combining supervised and unsupervised models for a given task. The idea behind BGCM is to cast the ensembling task as an optimization problem on a bipartite graph, where the objective function favors the smoothness of the prediction over the graph, as well as penalizing deviations from the initial labeling provided by supervised models. The authors propose to consolidate a classification solution by maximizing the consensus among both supervised predictions and unsupervised constraints. They show that their algorithm outperforms the component models on ten out of eleven classification tasks across three different datasets.

The Mixtures of Experts (ME) is another state-of-the-art ensembling algorithm (Jacobs et al., 1991). The ME algorithm has a close to our SWAF algorithm in terms of the underlying intuition of leveraging systems that are good at certain types of instances of a task. In this method, the problem space is partitioned stochastically into a number of sub-spaces and the idea is that the experts or learners are specialized on each subspace. ME uses divide-and-conquer principle to soft switch between learners covering different sub-spaces of the input. This method uses a supervised gating network which can be learnt using Expectation-Maximization (EM). More recently, Eigen et al. (2013) extended the ME to use a different gating network at each layer in a multilayer network, forming a Deep Mixture of Experts. By associating each input with a combination of experts at each layer, their model used different subsets of its units for different inputs, making it large as well as efficient.

3 Completed Work

In this section, we discuss our publications related to this proposal, and emphasize our major contributions while comparing to other existing work in the field.

3.1 Stacked Ensembles of Information Extractors for KBP (Rajani et al., 2015)

Many end-to-end information extraction systems are built using several underlying models that target specific sub-problems and are combined using ensembling. 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. As input, an ensemble takes the output of individual systems 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. For example, the bootstrapped self-training model by Angeli et al. (2015) uses pattern based methods for improving precision as well as distant supervision for improving recall in relation extraction. These models are then combined using voting to obtain the final output. However, voting is not very rewarding as it ignores the subtleties of the underlying models. We therefore proposed a more intelligent approach to ensembling component models that uses stacking as a way of ensembling information extractors.

3.1.1 Stacking

Stacking is a popular ensembling methodology in machine learning (Wolpert, 1992) and has been very successful 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. As for the meta-classifier, 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). In a final *post-processing* step, the slot fills that get classified as "correct" by the meta-classifier are kept while the others are discarded.

3.1.2 Using Provenance

Each system that extracts a slot-fill for a query, also provides provenance information about the fill. The filler provenance localizes the extracted slot-fill in a corpus of documents. 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. Where a slot-fill was extracted from can tell a lot about whether it is potentially correct or incorrect. This motivated us to use the provenance information as a feature to the stacker.

We compute two types of provenance scores, first using the *docid* information, and second using the *offset* information. The document-based provenance score is defined as follows. 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 agrees 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 answer have the same *docid* for their document provenance, then the offset provenance (OP) score for a slot fill by a system x is calculated as follows:

$$PO(n) = \frac{1}{|N|} \times \sum_{i \in N, i \neq n} \frac{|\text{substring(i)} \cap \text{substring(n)}|}{|\text{substring(i)} \cup \text{substring(n)}|}$$

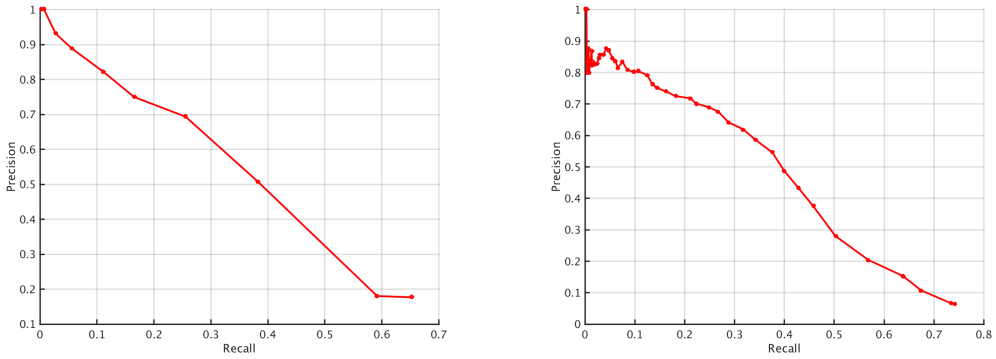
As per our definition, systems that extract slot fills from *different* documents for the same query slot have zero overlap among offsets. We note that the offset provenance is always used along with the document provenance and is thus useful in discriminating between slot fills extracted from the same 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. 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.

3.1.3 Eliminating Slot-Filler Aliases

When combining the output of different SF systems, it is possible that some slot-filler entities might overlap with each other. A system *A* could extract a filler F_1 for a slot *S* while another system *B* 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) and not combined into one, the precision of the final system is penalized for producing duplicate outputs.

In order to eliminate aliases from the output of ensembled system, we employ a technique derived by inverting the scheme used by Roth et al. (2013) for query expansion. The authors use a Wikipedia anchor-text model (Roth and Klakow, 2010) to generate aliases for given query entities. By including aliases for query names, the systems 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) 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.

3.1.4 Experimental Evaluation



Common systems dataset All 2014 SFV systems dataset

Figure 6: Precision-Recall curves for identifying the best voting performance on the two datasets

This section describes a comprehensive set of experiments evaluating the supervised ensembling approach on the KBP Engish Slot Filling (ESF) task. Our experiments are divided into two subsets based on the datasets they employ. Since our stacking approach relies on historical training data, we used the 2013 SFV data for training and build a dataset of one run for every team that participated in *both* the 2013 and 2014 competitions and called it the *common systems dataset*. There were 10 common teams of the 17 teams that participated in ESF 2014. The other dataset comprised 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.

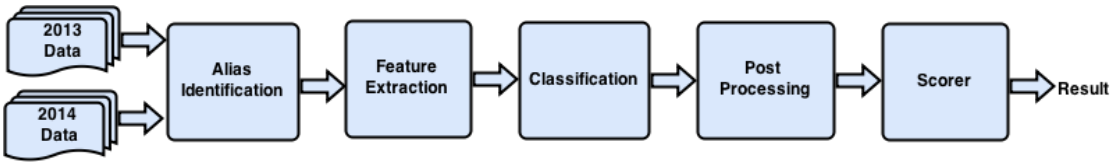


Figure 7: Our system pipeline for evaluating the supervised ensembling approache

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

We compared our stacking approach to voting and union ensembling baselines. Since both voting and union are unsupervised ensembling baselines, we evaluated on both the common systems dataset as well as the entire 2014 SFV dataset. The *Union* takes the combination of values for all

Approach	Precision	Recall	F1
Union	0.176	**0.647**	0.277
Best ESF system in 2014 (Stanford)	0.585	0.298	0.395
Voting (threshold learned on 2013 data)	**0.694**	0.256	0.374
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) + Relation	0.653	0.400	0.496
Stacking + Provenance (document and offset) + Relation	0.541	0.466	**0.501**

Table 2: Performance on the common systems dataset (10 systems) for various configurations. All approaches except the Stanford system are our implementations.

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 else if multiple systems attempt, then it only includes the slot fill with the highest confidence value as correct and discards the rest. For the *Voting* 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, thereafter, we use this threshold for the voting baseline on 2014 SFV dataset. The third baseline we compare to is an "oracle threshold" version of *Voting*. We do the same thing we did for 2013 dataset for the common systems dataset. However, 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. Figure 6 shows the 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 all 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 could not be predicted without using the 2014 dataset, this baseline has an unfair advantage. Table 1 shows the performance of all 3 baselines on the all 2014 SFV systems dataset.

We performed various ablations of the stacking algorithm by training on the 2013 data and testing on the 2014 data for the common systems. The first approach only used the confidence scores of the underlying systems as input for classifying an instance. The second approach used the slot type along with the confidence scores. The KBP slot-filling task had approximately 40 slot types for each query. This allowed the system to learn different evidence-combining functions for different slots if the classifier found that useful. Our third approach included document provenance feature and our fourth approach used the offset provenance in addition to the other features. We used for the L1-regularized SVM with a linear kernel (other classifiers gave similar results) for training. Figure 7 shows our system pipeline for evaluating supervised ensembling approaches. Table 2 gives the performance of all our supervised approaches as well as our unsupervised baselines for the common systems dataset.

Systems change from one year to another and training on previous year's data might not be

very intuitive. In order to have a better understanding of this, we plot a learning curve by training on different sizes of the 2013 data and then evaluating on the 2014 data for the common systems. Figure 8 shows the learning curve thus obtained. Although there are certain proportions 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.

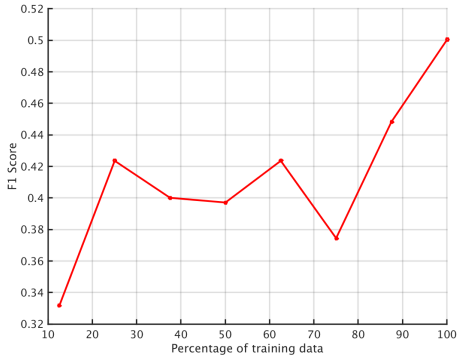


Figure 8: Learning curve for training on 2013 and testing on 2014 common systems dataset

Our results indicated that stacking with provenance information and slot type gives the best performance. Our stacking approach that used the 10 systems common between 2013 and 2014 also outperformed the ensembling baselines that had the advantage of using *all 65* of the 2014 systems. Our stacking approach would have presumably performed even better if we had access to 2013 training data for all 2014 systems. Of course, the best-performing 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 beats the best baseline for both the datasets by at least 6 F1 points using both the official and unofficial 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.

3.2 Stacking With Auxiliary Features (Rajani and Mooney, 2016a)

This paper was an extension of our previous work to generalize the idea of stacking with provenance features to other tasks. In this paper we introduced Stacking With Auxiliary Features (SWAF) that learns to fuse additional relevant information from multiple component systems as well as input instances to improve performance. We used two types of auxiliary features – *instance features* and *provenance features*. Instance features enable the stacker to discriminate across input instances while provenance features enable the stacker to discriminate across component systems. We demonstrated our approach on three very different and difficult problems: Cold Start Slot Filling, Tri-lingual Entity Discovery and Linking, and ImageNet Object Detection. We obtained new

state-of-the-art results on the first two tasks and significant improvements on the ImageNet task, thus verifying the power and generality of our approach. Figure 9 shows an overview of our system which trains a final meta-classifier for combining multiple systems. The specific auxiliary features depend on the task under consideration as described in the following sections.

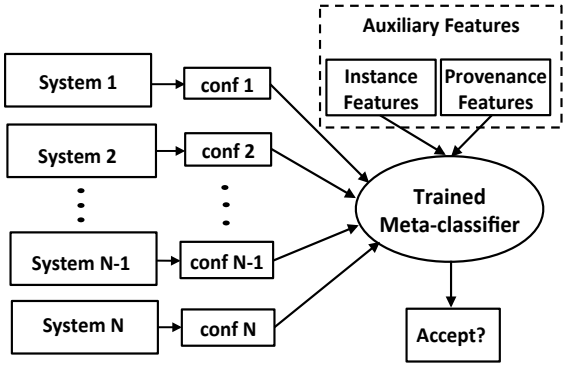


Figure 9: Our SWAF approach to combining system outputs using confidence scores and two types of auxiliary features for improving prediction.

3.2.1 Stacking

By training on a set of supervised data, disjoint from that used to train the individual models, stacking learns how to combine their results into an improved ensemble model that performs better than each individual component system. In our approach to stacking, the output is represented as a *key-value* pair. The meta-classifier makes a binary decision for each distinct output pair. Thus before deploying the algorithm on a task, it is crucial to identify the *key* in the task output which serves as a unique handle for ensembling across systems as well as the *values* which are results provided by a system for a particular key. Note that there is only one instance of a key in a given system's output, while there could be multiple values for a given key. The output of the ensembling system is similar to the output of an individual system, but it productively aggregates results from different systems. In a final *post-processing* step, the outputs that get classified as "correct" by the classifier are kept while the rest are discarded.

For the CSSF task, the *key* is a *query entity* along with a slot type, such as per:age of "Barack Obama", and the *value* is a computed *slot fill*, such as "55". For list-valued slot types such as org:subsidiaries, the key instance is repeated in the output for each value. For TEDL, we define the key to be the *KB (or NIL) ID* of an entity and the value to be a *mention*, i.e. a string that references that entity in the text. For ImageNet Object Detection, we represent the image ID as the *key* and the *value* is a detected object category. The next step is to represent the output pair instances consistently. For a particular *key-value* pair, if a system produced it and if it also provides a confidence score then we use that as input but if it doesn't provide a confidence value then we assume it to be 1.0. On the other hand, if a system did not produce a *key-value* pair then we use a confidence of zero i.e. that *key-value* is incorrect according to that system. The confidence for each

system is then given as input to the meta-classifier together with the auxiliary features described in the next section.

3.2.2 Auxiliary Features

For the provenance features, we use provenance information provided by systems for each generated output pair. Provenance indicates the origin or source of the generated output and thus depends on the task. For the KBP tasks, provenance is in the form of a substring in the document corpus. For object detection, provenance is in the form of a bounding box in the image. For CSSF, if a system successfully extracts a relation then it must provide a slot fill provenance indicating the location of the extracted slot fill in the corpus. For TEDL, if a system successfully links a mention to a KB ID then it must provide the mention provenance indicating the origin of the mention in the corpus. For both these NLP tasks, the provenance is in the form of *docid* and *startoffset–endoffset* that gives the source document in the corpus and offset in this document. For the ImageNet object detection task, if a system successfully detects a target category then it must provide the object bounding box localizing the object in the image. The bounding box is in the form of $\langle x_{min}, y_{min}, x_{max}, y_{max} \rangle$. The bounding box for object detection is similar to provenance for the KBP tasks, giving *where* in the input is the information supporting the conclusion.

For the CSSF and TEDL tasks, Jaccard similarity is used to capture the provenance information between systems, just like in our previous paper (Rajani et al., 2015). On the other hand, for the object detection task, the Jaccard coefficient is used to measure the overlap between bounding boxes across systems as follows. For a given image ID, if N systems detect the same object instance, then the bounding box overlap (BBO) score for a system n is calculated as the the intersection of the areas of bounding boxes, divided by their union:

$$BBO(n) = \frac{1}{|N|} \times \sum_{i \in N, i \neq n} \frac{|\mathsf{Area(i)} \cap \mathsf{Area(n)}|}{|\mathsf{Area(i)} \cup \mathsf{Area(n)}|}$$

We note that for the CSSF task, two systems are said to have extracted the same slot fill for a *key* if the fills are exactly same, on the other hand, for the TEDL task, two systems are said to have linked the same mention for a *key* if the mentions overlap to any extent. Finally for the ImageNet task, two systems are said to have detected the same object instance for a *key* if the Intersection Over Union (IOU) of the areas of their bounding boxes is greater than 0.5. If the output *values* don't meet this criteria for a given *key*, then they are considered to be two different values for the same key. For a *key-value* pair, we obtain as many features as the number of component systems using the above equations for each task.

The idea behind using the *instance auxiliary features* is that some systems are better at some sub-tasks. One could imagine that some systems are good at extracting relations for certain slot types (e.g.: slot types that expect a location as an output) and similarly some models learn to better localize object categories with certain attributes (e.g.: furry objects). Such information if available at the time of classification could be a deal breaker. For example, the stacker could learn not to trust an object detection output for target class 'dog' from a system that is not good at detecting dogs. The instance features enable the stacker to learn such patterns using task specific information in conjunction with the provenance features. For the CSSF task, we use the one-hot encoding of the slot type (e.g. per:age) and for the TEDL task, we use the one-hot encoding of entity type (PER/ORG/GPE/FAC/LOC) as instance features. Another instance auxiliary feature

we used for the KBP tasks is the similarity between the *key* document and the *value* document. For the CSSF task, the *key* is the query entity along with slot type and thus the *key* document is the query document provided to participants to disambiguate query entities that could potentially have the same name but refer to different entities. For the TEDL task, the *key* is the KB ID of an entity and so we created a pseudo-document for every entity consisting of it's KB description as well as all relations involving the entity that exist in the KB, which we use as the *key* document. The document that the CSSF and TEDL systems provide as provenance is the *value* document for both the tasks. So the instance auxiliary feature uses cosine similarity to compare the *key* and *value* documents represented as standard TF-IDF weighted vectors. The intuition is that if a *key-value* pair is correct then it would have high syntactic overlap between the *key* and *value* documents because they contain discourse on the same entity.

For the ImageNet task, we use two types of instance auxiliary features. Firstly, we use the one-hot encoding of the object categories (total 200) as a feature to the stacker. Secondly, we use the bag of visual words for each image as a instance feature to the stacker. The bag of visual words use Scale-Invariant Feature Transform (SIFT) as the feature descriptor (Lowe, 2004). Note that some underlying object detection systems also use bag of visual words for classification and we show that using these for learning the top-level meta-classifier further boosts the performance.

3.2.3 Post-processing

Once we obtain the decisions on each of the key-value pairs from the stacker, we perform some final post-processing to produce output that is in the same format as generated by a single system. For CSSF, each list-valued slot fill that is classified as correct is included in the final output. For single-valued slots, if multiple fills are classified as correct for the same query and slot type, we include the fill with the highest meta-classifier confidence. For the TEDL task, for each entity mention link that is classified as correct, if the link is a KB cluster ID then we include it in the final output, but if the link is a NIL cluster ID then we keep it aside until all mention links are processed. Thereafter, we resolve the NIL IDs across systems since NIL ID's for each system are unique. We merge NIL clusters across systems into one if there is at least one common entity mention among them. Finally, we give a new NIL ID for these newly merged clusters.

For the ImageNet object detection task, each object instance that is classified as correct by the stacker is included in the final output and its bounding box is calculated as follows. If multiple systems detected the object instance, then we sum the overlapping areas between a system's bounding box and that of every *other* system that also detected the exact same instance and we do this for every such system. The system with the *maximum* sum has a bounding box with the maximum overlapping area with other systems, and thus is used as the bounding box for the ensemble. Note that in case of two systems, this method is redundant and we include the bounding box produced by the system with a higher confidence score. We also experimented with using the *union* or the *intersection* of bounding boxes across systems as the aggregate bounding box for the ensemble but were heavily penalized due to the following evaluation metric used by the scorer. For a standard sized image (larger than 25×25 pixels), the ImageNet detection scorer considers an object to be localized correctly in an image if the IOU of the output bounding box and the ground-truth bounding box is ≥ 0.5.

3.2.4 Experimental Evaluation

All KBP results were obtained using the official NIST scorer provided after the competition ended.[5] For object detection, we used the scorer provided with the ImageNet devkit.

For the two KBP tasks, we only use systems that participated in both the 2014 and 2015 competitions. This allows us to train the stacker on 2014 system outputs and use the trained model to evaluate on the 2015 data. In this way, we used 10 common systems for CSSF and 6 for TEDL. We were unable to obtain the competing systems' outputs for the ImageNet task, so we used two state-of-the-art deep neural models pre-trained on the ImageNet object-detection training set, the ZF and VGG models described in (Ren et al., 2015). We ran these models on the validation set using the faster-RCNN method (Ren et al., 2015) with selective search (Uijlings et al., 2013) using the Caffe system (Jia et al., 2014). We also use the Deformable Parts Model (DPM) (Felzenszwalb et al., 2010) with selective search for object detection, to produce a final ensemble of three systems. DPM is very slow to test and was unable to process the entire test set on all 200 categories. Therefore, we performed 10-fold cross validation on the validation set, testing our approach on a total of about 20K images.

Method	Precision	Recall	F1
Stacking with both provenance + instance auxiliary features	0.466	**0.331**	**0.387**
Stacking with just provenance auxiliary features	**0.508**	0.286	0.366
Stacking with just instance auxiliary features	0.498	0.284	0.360
Stacking without auxiliary features	0.497	0.282	0.359
Top ranked CSSF system in 2015 (Angeli et al., 2015)	0.399	0.306	0.346
Oracle Voting baseline (3 or more systems must agree)	0.438	0.272	0.336
Mixtures of Experts (ME) model	0.479	0.184	0.266

Table 3: Results on 2015 Cold Start Slot Filling (CSSF) task using the official NIST scorer

Method	Precision	Recall	F1
Stacking with both provenance + instance auxiliary features	**0.814**	0.515	**0.630**
Stacking with just provenance auxiliary features	0.814	0.508	0.625
Stacking with just instance auxiliary features	0.783	0.511	0.619
Stacking without auxiliary features	0.729	0.528	0.613
Top ranked TEDL system in 2015 (Sil et al., 2015)	0.693	0.547	0.611
Mixtures of Experts (ME) model	0.721	0.494	0.587
Oracle Voting baseline (4 or more systems must agree)	0.514	**0.601**	0.554

Table 4: Results on 2015 Tri-lingual Entity Discovery and Linking (TEDL) task using the official NIST scorer and the CEAFm metric

For the CSSF task, systems are evaluated against a gold standard using precision, recall and F1 scores using the extracted slot fills. The TEDL evaluation uses the mention CEAF metric (Ji et

[5] http://www.nist.gov/tac/2015/KBP/ColdStart/tools.html, https://github.com/wikilinks/neleval

Method	Mean AP	Median AP
Stacking with provenance + instance auxiliary features	**0.506**	**0.497**
Stacking with just provenance auxiliary features	0.502	0.494
Mixtures of experts (ME) model	0.494	0.489
Stacking with just instance features	0.461	0.450
Stacking without auxiliary features	0.451	0.441
Best standalone system (VGG + selective search)	0.434	0.430
Oracle Voting baseline (1 or more systems must agree)	0.366	0.368

Table 5: Results on 2015 ImageNet object detection task using the official ImageNet scorer.

al., 2015) for measuring precision, recall and F1. This metric finds the optimal alignment between system and gold standard clusters, and then evaluates precision and recall micro-averaged over mentions. For the ImageNet challenge, the detection task is evaluated using the average precision (AP) on a precision/recall curve. The predicted bounding box for a detection is considered correct if its intersection over union with the ground truth exceeds a threshold of 0.5 (Russakovsky et al., 2015). The official scorer gives the AP for each of the 200 classes along with the median and mean AP. We report the median AP and mean AP (mAP).

We compare our results to several baselines. For all three tasks, we compare to stacking without using any auxiliary features and various ablations of the provenance and instance auxiliary features. We also compare to the top ranked systems for both the CSSF and TEDL tasks in 2015. For the 2015 object detection task, we were unable to run the state-of-the-art system and thus it was not part of our ensemble. For this reason, only for this task, we compare our results to the best performing *component* system. Our results also include the "oracle" voting baseline for ensembling the system outputs. For this approach, we vary the threshold on the number of systems that must agree to identify an "oracle" threshold that results in the highest F1 score by plotting a Precision-Recall curve and finding the best F1. This method found an optimal threshold of 3 for CSSF, 4 for TEDL and 1 for the object detection task. We note that oracle voting is "cheating" to give the best possible voting baseline. We also compare our results to a state-of-the-art ensembling approach called Mixtures of Experts (ME) (Jacobs et al., 1991) because of its proximity to SWAF in terms of the underlying intuition. In this method, the problem space is partitioned stochastically into a number of sub-spaces and the idea is that the experts or learners are specialized on each subspace. ME uses divide-and-conquer principle to soft switch between learners covering different sub-spaces of the input. This method uses a supervised gating network which can be learnt using Expectation-Maximization (EM). Recollect that the intuition behind using the instance features is also very similar.

Tables 3 and 4 show the results for CSSF and TEDL tasks, respectively. Stacking with both instance and provenance features performed consistently the best on both the tasks beating all ablations of the auxiliary features, the ME algorithm as well as the top ranked systems for 2015. Oracle voting performs very poorly, indicating that naive ensembling is not advantageous. The relative ranking of our approaches is same on both the CSSF and TEDL tasks, thus demonstrating that our approach is very general and provides improved performance on two quite different and challenging NLP problems. On the other hand, the ME algorithm is not robust and we observe that it's performance is inversely proportional to the number of component systems. Since the CSSF

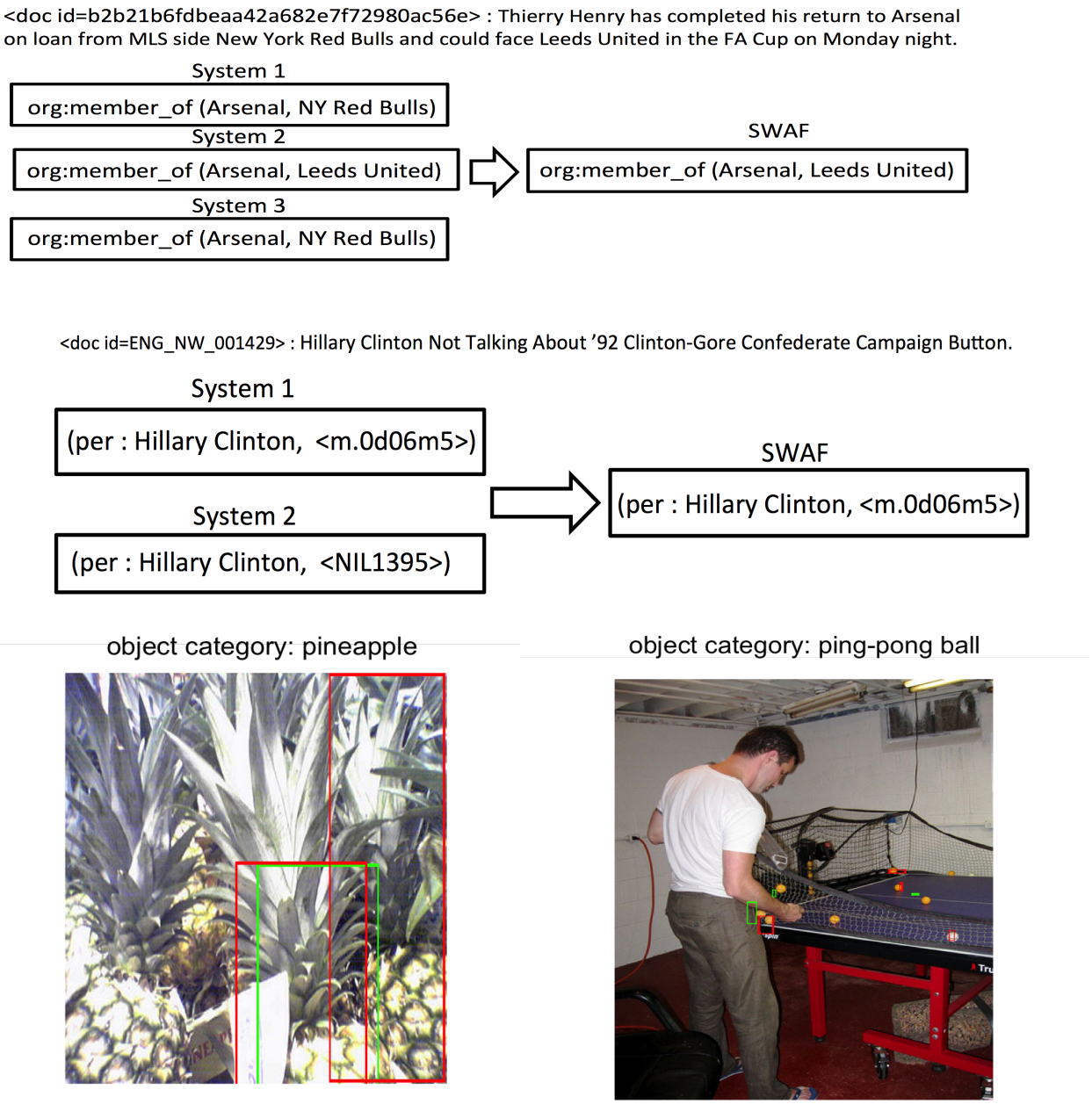


Figure 10: Sample outputs obtained using SWAF on various 2015 tasks. The top row shows output from the CSSF KBP task while in the middle is the TEDL task. The text snippet above both examples displays the provenance used by SWAF while classifying. The bottom row is output obtained on the ImageNet object detection task. The green bounding box are those obtained by the systems and among those, the red ones are classified correct by SWAF.

task had more systems (i.e. 10) than the TEDL task (i.e. 6), the ME algorithm is unable to learn a good gating network for system selection and thus performs worse than the voting baseline as well. Based on the results, it is observed that the SWAF algorithm is better at improving precision than recall, and the reason is because we only choose systems that are common between years which

leads to a small number of component systems. Since TEDL has even smaller number of systems than CSSF, the recall is much worse. By training on a small percentage of previous year's data incrementally and testing on entire 2015 data at every step, we concluded that although systems get better over the years, it is still advantageous to train on previous year's output. For both the NLP tasks, we used L1 regularized linear SVM weighted by the number of instances for each class, as a meta-classifier for stacking. A small random sample of the training set (10%) was used as validation to set these parameters.

Table 5 shows the results for the ImageNet 2015 object detection task. Again, using stacking with both types of auxiliary features beats the best individual component system as well as the oracle voting baseline significantly. For the voting baseline, we consider an object instance to be the same if the systems' bounding boxes have IOU greater than 0.5. If we were able to use the top-ranked system from the competition, we could have potentially obtained state-of-the-art results. Since we use cross-validation for obtaining the results, we performed the pairwise 1-tailed test for statistical significance with significance level 0.05 and found that using any ablation of stacking with auxiliary features is statistically significantly better than using the best component system, with $p < 0.05$, although *just* using stacking is not statistically significant. The ME algorithm performs significantly better than the individual component because of the fewer component systems (i.e. 3) but is still worse than our best approach. For this task, we used SVM with the RBF kernel as the meta-classifier for stacking. On analyzing the results, we found that the AP of several object classes differed widely across systems and even more so between the deep systems and DPM. Using SWAF, the meta-classifier learns to discriminate systems based on the auxiliary features and is thus able to leverage the best aspects of each individual system. Based on the outputs obtained from SWAF, we found that our approach particularly does well on localizing objects in images that have multiple instances of the *same* object, i.e. the image can be considered to be "cluttered". Figure 10 shows a random sample of correct results obtained by our best approach for each of the three tasks.

3.3 Combining Supervised and Unsupervised Ensembles for Knowledge Base Population (Rajani and Mooney, 2016b)

In our approach to using SWAF for various tasks, we were only able to use shared systems that had historical training data. However, in some situations, we would like to ensemble systems for which we have no historical performance data. For example, due to privacy, some companies or agencies may be willing to share their final models or meta-level model output but not the raw data itself. Simple methods such as voting permit "unsupervised" ensembling but fail to exploit supervision for those systems for which we *do* have training data. Therefore, we presented an approach that utilizes supervised *and* unsupervised ensembling to exploit the advantages of both. We first used unsupervised ensembling to combine systems without training data, and then used stacking to combine this ensembled system with other systems for which training data is available.

Using this new approach, we demonstrated new state-of-the-art results on two separate tasks on *Cold Start Slot-Filling* (CSSF)[6] and the *Tri-lingual Entity Discovery and Linking* (TEDL) (Ji et al., 2015). Our approach outperformed the best individual system as well as other ensembling methods such as stacking only the shared systems which was our previous supervised approach

[6] `http://www.nist.gov/tac/2015/KBP/ColdStart/guidelines.html`

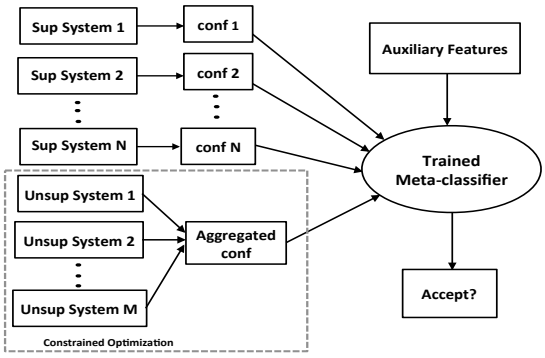


Figure 11: Illustration of our approach to combine supervised and unsupervised ensembles.

(Rajani and Mooney, 2016a). As part of this work, we also proposed two new auxiliary features for the CSSF and TEDL tasks and verified that incorporating them in the combined approach improved performance. Figure 11 illustrates our system which trains a final meta-classifier for combining multiple systems using confidence scores and other auxiliary features depending on the task.

3.3.1 Unsupervised Ensembling Approach

Only 38 of the 70 systems that participated in CSSF 2015 also participated in 2014, and only 24 of the 34 systems that participated in TEDL 2015 also participated in the 2014 EDL task. Therefore, many KBP systems in 2015 were new and did not have past training data needed for the supervised approach. In fact, some of the new systems performed better than the shared systems, for example the *hltcoe* system did not participate in 2014 but was ranked 4^{th} in the 2015 TEDL task (Ji et al., 2015). We first ensembled these unsupervised systems using the constrained optimization approach described by Wang et al. (2013). Their approach is specific to the English slot-filling task and also relies a bit on past data for identifying certain parameter values. Below we describe our modifications to their approach so that it can be applied to both KBP tasks in a purely unsupervised manner. The bottom half of Figure 11 shows the ensembling of the systems without historical training data.

The approach in Wang et al. (2013) aggregates the raw confidence values produced by individual KBP systems to arrive at a single aggregated confidence value for each key-value. Suppose that $V_1, \ldots, V_M$ are the M distinct values produced by the systems and N_i is the number of times the value V_i is produced by the systems. Then Wang et al. (2013) produce an aggregated confidence by solving the following optimization problem:

$$\min_{0 \leq x_i \leq 1} \sum_{i=1}^{M} \sum_{j=1}^{N_i} w_{ij} \left(x_i - c_i\left(j\right)\right)^2,$$

where c_i denotes the raw confidence score and x_i denotes the aggregated confidence score for V_i, $w_{ij} \geq 0$ is a non-negative weight assigned to each instance. Equation 3.3.1 ensures that the aggregated confidence score is close to the raw score as well as proportional to the agreement

among systems on a value for a given key. Thus for a given key, if a system's value is also produced by multiple other systems, it would have a higher score than if it were not produced by any other system. The authors used the inverse ranking of the average precision previously achieved by individual systems as the weights in the above equation. However since we used this approach for systems that did not have historical data, we used uniform weights across all unsupervised systems for both the tasks.

Equation 3.3.1 is subject to certain constraints on the confidence values depending on the task. For the slot-filling task, the authors define two different constraints based on whether the slot type is single valued or list valued. For single-valued slot types, only one slot value can be correct and thus the constraint is based on the mutual exclusion property of the slot values:

$$P(V_1) + P(V_2) + \cdots + P(V_M) \leq 1$$

This constraint allows only one of the slot values to have a substantially higher probability compared to rest. On the other hand, for list-valued slot types, the 1 in the above equation is replaced by the value n_c/n where n_c is the average number of correct slot fills for that slot type across all entities in the previous year and n is the total number of slot fills for that slot type across all entities. This approach to estimating the number of correct values can be thought of as *collective precision* for the slot type achieved by the set of systems. For the newly introduced slot inverses in 2015, we used the same ratio as that of the corresponding original slot type. Thus the slot type *per:parents* (new slot type) would have the same ratio as that of *per:children*.

For the TEDL task, we used the KB ID as the key and thus use the entity type for defining the constraint on the values. For each of the entity types (PER, ORG and GPE) we replaced the quantity on the right hand side in Equation 3.3.1 by the ratio of the average number of correct values for that entity type in 2014 to the total number of values for that entity type, across all entities. For the two new entity types introduced in 2015 (FAC and LOC), we used the same ratio as that of GPE because of their semantic similarities.

The output from this approach for both tasks is a set of key-values with aggregated confidence scores across all unsupervised systems which go directly into the stacker as shown in Figure 11. Using the aggregation approach as opposed to directly using the raw confidence scores allows the classifier to meaningfully compare confidence scores across multiple systems although they are produced by very diverse systems. Another technique for unsupervised ensembling that we experimented with in place of the constrained optimization approach is the Bipartite Graph based Consensus Maximization (BGCM) approach by Gao et al. (2009) which was discussed in Section 2.4.

3.3.2 Combining Supervised and Unsupervised

For the KBP systems that were common between years, we used the stacking method discussed in Section 3.2.1. The top half of Figure 11 illustrates ensembling multiple systems with historical training data using the supervised SWAF approach. We proposed to combine the supervised and unsupervised methods using a stacked meta-classifier as the final arbiter for accepting a given key-value. The outputs from the supervised and unsupervised systems are fed into the stacker in a consistent format such that there is a unique input *key-value* pair. Most KBP teams submit multiple variations of their system. Before ensembling, we first combine multiple runs of the same team into one. Thus, for CSSF we obtained 10 systems (one for each team) for which we have supervised

data for training stacking and 13 systems for which we used unsupervised ensembling. Similarly, we obtained6 teams that had 2014 training data and 4 teams that did not have training data for the TEDL task. Thus using the notation in Figure 11, for TEDL, $N = 6$ and $M = 4$ while for CSSF, $N = 10$ and $M = 13$.

The unsupervised method produced aggregated, calibrated confidence scores which go directly into our final meta-classifier. We treat this combination as a single system called the *unsupervised ensemble*. We add the unsupervised ensemble as an additional system to the stacker, thus giving us a total of $N + 1$, that is 11 CSSF and 7 TEDL systems. Once we have extracted the auxiliary features for each of the N supervised systems and the unsupervised ensemble for both years, we train the stacker on 2014 systems, and test on the 2015 systems. The unsupervised ensemble for each year is composed of different systems, but hopefully the stacker learns to combine a generic unsupervised ensemble with the supervised systems that are shared across years. This allows the stacker to arbitrate the final correctness of a key-value pair, combining systems for which we have no historical data with systems for which training data *is* available. To learn the meta-classifier, we use an L1-regularized SVM with a linear kernel (Fan et al., 2008) (other classifiers gave similar results).

Once we obtain the decisions from the stacker on every output instance, we perform some post-processing so that the aggregated output is consistent and appears to be from a single system. All the instances classified as correct by the classifier are kept while those that are classified as incorrect are discarded. The correct instances are then processed based on the task at hand and the process if exactly the same as described previously in Section 3.2.3

3.3.3 New Auxiliary Features

The 2015 CSSF task had a much smaller corpus of shorter documents compared to the previous year's slot-filling corpus (Ellis et al., 2015; Surdeanu and Ji, 2014). Thus, the provenance feature of Rajani et al. (2015) did not sufficiently capture the reliability of a slot fill based on where it was extracted. Slot filling queries were provided to participants in an XML format that included the query entity's ID, name, entity type, the document where the entity appears, and beginning and end offsets in the document where that entity appears. This allowed the participants to disambiguate query entities that could potentially have the same name but refer to different entities. Below is a sample query from the 2015 task:

```
<query id="CSSF15_ENG_0006e06ebf">
  <name>Walmart</name>
  <docid>ad4358e0c4c18e472c13bbc27a6b7ca5</docid>
  <beg>232</beg>
  <end>238</end>
  <enttype>org</enttype>
  <slot0>org:date_dissolved</slot0>
</query>
```

The <docid> tag refers to the document where the query entity appears, which we will call the *query document*. Our first new feature involved measuring the similarity between this query document and the provenance document that is provided by a given system. We represent the query and provenance documents as standard TF-IDF weighted vectors and use cosine similarity to compare

documents. Therefore every system that provides a slot fill, also provides the provenance for the fill and thus has a similarity score with the query document. If a system does not provide a particular slot fill then its document similarity score is simply zero. This feature is intended to measure the degree to which the system's provenance document is referencing the correct query entity.

Our second new feature measured the document similarity between the *provenance documents* that different systems provide. Suppose for a given query and slot type, n systems provide the same slot fill. For each of the n systems, we measure the average document cosine similarity between the system's provenance document and those of the other $n-1$ systems. The previous approach simply measured whether systems agreed on the exact provenance document. By softening this to take into account the *similarity* of provenance documents, we hope to more flexibly measure provenance agreement between systems. We use these similarity features for the TEDL tasks as well, except that the TEDL task does not have a query document and so we use the FreeBase entry for the mention. We create a *pseudo* document by appending the FreeBase definition, description and relations of the query entity. In this way the new auxiliary feature captures similarity between the provenance document and the *pseudo* FreeBase document.

3.3.4 Experimental Evaluation

Methodology	Precision	Recall	F1
Combined stacking and constrained optimization	0.4679	**0.4314**	**0.4489**
Top ranked SFV system in 2015 (Rodriguez et al., 2015)	0.4930	0.3910	0.4361
Stacking using BGCM instead of constrained optimization	**0.5901**	0.3021	0.3996
BGCM for combining supervised and unsupervised systems	0.4902	0.3363	0.3989
SWAF approach (Rajani and Mooney, 2016a)	0.4656	0.3312	0.3871
Top ranked CSSF system in 2015 (Angeli et al., 2015)	0.3989	0.3058	0.3462
Oracle Voting baseline (3 or more systems must agree)	0.4384	0.2720	0.3357
Constrained optimization approach (Wang et al., 2013)	0.1712	0.3998	0.2397

Table 6: Results on 2015 Cold Start Slot Filling (CSSF) task using the official NIST scorer

Methodology	Precision	Recall	F1
Combined stacking and constrained optimization	0.686	**0.624**	**0.653**
Stacking using BGCM instead of constrained optimization	0.803	0.525	0.635
BGCM for combining supervised and unsupervised outputs	0.810	0.517	0.631
SWAF approach (Rajani and Mooney, 2016a)	0.813	0.515	0.630
Top ranked TEDL system in 2015 (Sil et al., 2015)	0.693	0.547	0.611
Oracle Voting baseline (4 or more systems must agree)	0.514	0.601	0.554
Constrained optimization approach (Wang et al., 2013)	0.445	0.176	0.252

Table 7: Results on 2015 Tri-lingual Entity Discovery and Linking (TEDL) using official NIST scorer and CEAF metric

Tables 6 and 7 show CSSF and TEDL results. Our full system, which combines supervised and unsupervised ensembling performed the best on both tasks. TAC-KBP also includes the Slot Filler

Validation (SFV) task[7] where the goal is to ensemble/filter outputs from multiple slot filling systems. The top ranked system in 2015 (Rodriguez et al., 2015) does substantially better than many of the other ensembling approaches, but it does not do as well as our best performing system. The purely supervised approach using auxiliary features of Rajani and Mooney (2016a) performs substantially worse, although still outperforming the top-ranked individual system in the 2015 competition. These approaches only use the common systems from 2014, thus ignoring approximately half of the systems. The approach of Wang et al. (2013) performs very poorly by itself; but when combined with stacking gives a boost to recall and thus the overall $F1$. Note that all our combined methods have a substantially higher recall. The oracle voting baseline also performs very poorly indicating that naive ensembling is not advantageous. For the TEDL task, the relative ranking of the approaches is similar to those obtained for CSSF, proving that our approach is very general and improves performance on two quite different and challenging problems.

Even though it is obvious that the boost in our recall was because of adding the unsupervised systems, it isn't clear how many new *key-value* pairs were generated by these systems. We thus evaluated the contribution of the systems ensembled using the supervised approach and those ensembled using the unsupervised approach, to the final combination for both the tasks. Figure 12 shows the number of unique as well as common *key-value* pairs that were contributed by each of the approaches. The unique pairs are those that were produced by one approach but not the other and the common pairs are those that were produced by both approaches. We found that approximately one third of the input pairs in the combination came from the unique pairs produced just by the unsupervised systems for both the TEDL and CSSF tasks. Only about 15% and 22% of the total input pairs were common between the two approaches for the TEDL and CSSF tasks respectively. Our findings highlight the importance of utilizing systems that do not have historical training data.

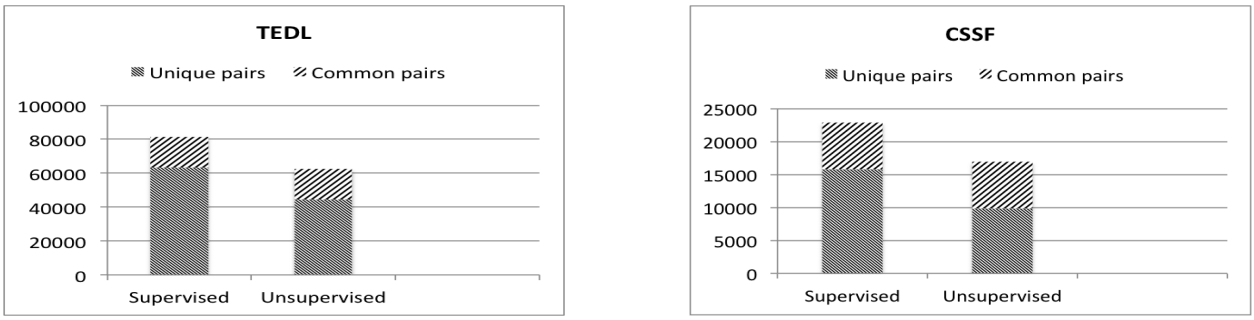


Figure 12: Total number of unique and common input pairs contributed by the supervised and unsupervised systems to the combination for the TEDL and CSSF tasks respectively.

[7] http://www.nist.gov/tac/2015/KBP/SFValidation/index.html

4 Proposed Work

In this section we describe the proposed work of further research. The proposed future work can be divided into two categories: short-term proposals, which will definitely be completed for the final dissertation, and long-term proposals, which are more ambitious future research goals that may not all be included in the thesis.

The short-term proposals involve identifying and consolidating auxiliary features that boost the performance of the ensemble. As mentioned earlier, the auxiliary features fall into two categories – instance features and provenance features. While most of the completed work focused on identifying and using the provenance features, the instance features also play an important role in the decision making process and will thus be a focus of the future short-term proposal.

The long-term proposal includes preliminary assessments on using explanations as an additional source of auxiliary features. So far the completed work relies on using provenance generated by component systems along with the output instance. This auxiliary feature thus captures the "where" of an output instance. The long-term proposal focuses on using the "why" of an output as auxiliary features. These features would capture the explanation for generating an output instance. Just like the completed work, the long-term proposals also target tasks that involve text and images. Another long-term proposal is to demonstrate our algorithm on the Visual Question Answering (VQA) task.

4.1 Short Term Proposals

In this section, we propose and discuss several additional experiments which should be completed for the final dissertation. These can be subdivided into experiments for generating new *instance* features and experiments for generating foreign language features.

4.1.1 Instance-level Semantic Features

As discussed in Section 3.2.2, *instance* features are those that capture information about the instance that could be useful at classification time. The completed work used only the superficial instance-level information such as slot type for the slot-filling task, the entity type for the entity-linking task and SIFT descriptors for the ImageNet object-detection task. The next step is to explore more comprehensive features that capture semantic context. For the EDL task, this could be semantic similarity between the mention document and the FreeBase *pseudo* document of the entity while for the CSSF task, the similarity is between the provenance document and the query document provided for disambiguation. For the ImageNet task, semantic context could be attributes of the image class.

Recently, Francis-Landau et al. (2016) used contextual information to disambiguate entity mentions using convolutional neural networks, for the entity-linking problem. The CNN computes similarities between a mention's source document context and its potential entity targets at multiple granularities. The authors integrate their model into an existing entity-linking system to achieve state-of-the-art performance on multiple entity linking datasets. In our experiments involving document similarity features, we considered the mention document and target document as a single piece of text. However, it has been shown that granularity of both the mention and target document matters (Francis-Landau et al., 2016). The authors compute convolutional semantic similarity at

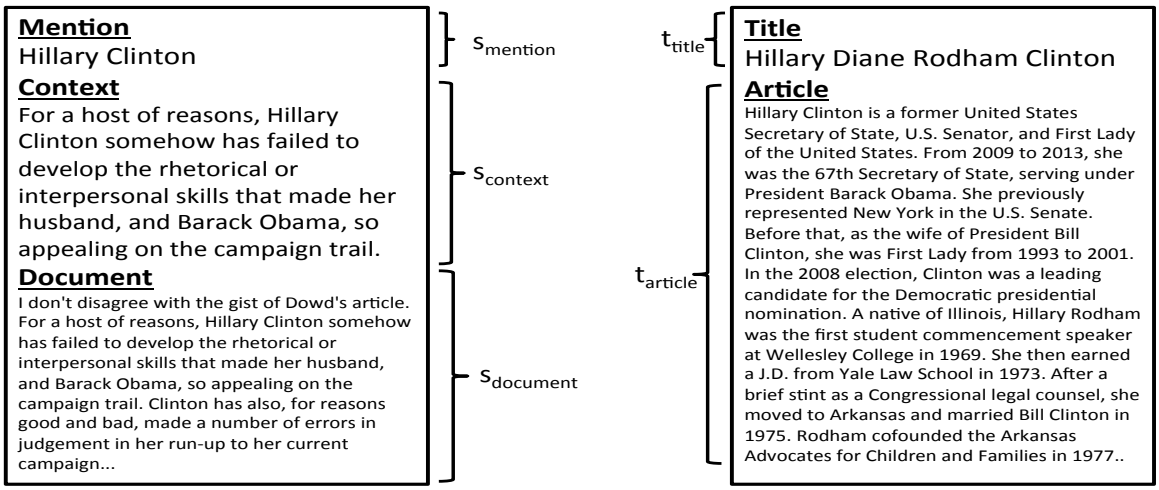


Figure 13: Example illustrating the granularities on the source and target text and the six similarity pairs between the source and target. The source is taken from the NIST KBP 2016 EDL competition and the target is the FreeBase entry for Hillary Clinton.

different granularities of the mention text and the potential target entity text. This results in six similarity pairs between the mention entity text represented as s and target entity text which in our case is a FreeBase KB entry represented as t. Figure 13 shows an example illustrating these pairs for a mention of *Hillary Clinton* and the KB entry for the same entity.

We plan on using these convolutional features for the two KBP tasks that involved a mention text and a target text. The mention text for the slot filling task is the document used for disambiguating the query entity and the target text is the text where the fill for the query entity is extracted. Recall that the auxiliary features used in completed work were either syntactic (what we call as provenance features) or nominal (what we call as instance features). Thus using the convolutional features would add the semantic dimension to the auxiliary features.

The ILSVRC data along with images and bounding boxes also provides an attributes dataset for some of the object categories (Russakovsky et al., 2015). The objects are annotated with the following attributes:

- **Color:** black, blue, brown, gray, green, orange, pink, red, violet, white, yellow
- **Pattern:** spotted, striped
- **Shape:** long, round, rectangular, square
- **Texture:** furry, smooth, rough, shiny, metallic, vegetation, wooden, wet

Such attributes give a more deeper semantic dimension to the object being localized. We think that these attributes as instance-level auxiliary features would enable the classifier to better discriminate across objects.

4.1.2 Foreign Language Features

In Section 3.2.4, we demonstrated stacking with auxiliary features on the slot-filling and entity-linking tasks for Spanish and Chinese. Although our ensemble beat the top ranked system in the

competition, the improvements on the foreign language datasets were very modest. For the slot-filling task, the training data did not include either of these foreign languages and thus the results are justifiable. However, for the entity-linking task, the training data included Spanish and Chinese along with English. This leads us to conclude that the auxiliary features used for English do not translate very well to the Spanish and Chinese domains. We would thus like to investigate the foreign language tasks in detail and design features that are not language specific for the KBP tasks. One straightforward feature that requires training data for all languages is to use the language indicator itself as a feature to learn if some systems are just better on some languages.

Recently, Sil and Florian (2016) proposed a one-for-all solution to the tri-lingual entity-linking task that trains a single model on English data and achieves state-of-the-art results on Spanish and Chinese along with English. Their work motivated us to use non-lexical auxiliary features so that they are language independent. Features related to KB inlinks, outlinks and redirects are one such example of non-lexical features. Other interesting features used by the authors are as follows:

- **Entity category PMI**: Calculates the PMI between pair of entities (e_1, e_2) that co-occur in a document for each mention.
- **Categorical relation frequency**: For a pair of entities (e, e') in a mention, count the number of KB relations that exist between them.
- **Title co-occurence frequency**: For every pair of consecutive entities (e, e'), computes the number of times e' appears as a link in the KB page for e, and vice versa (Cucerzan, 2007).

Our goal is to adopt some of these non-lexical features in our model for the KBP tasks using foreign language. We hope that such features would enable the classifier to learn beyond the superficial patterns in the dataset and obtain a more consistent performance across languages.

4.2 Long Term Proposals

Recently there has been some work from the deep learning community on generating explanations as a way to better understand and interpret the decisions made by deep neural networks (Antol et al., 2015; Goyal et al., 2016; Hendricks et al., 2016). We aim to use these explanations as auxiliary features to our model as a step towards the goal of explainable AI. We now describe using these as possible long-term research proposals which could be addressed in the final dissertation. These are more ambitious and require much more work than the short-term proposals. The first part of the longer term work falls into two categories – using text as explanation and using images as explanation. The second longer term work is to ensemble systems participating in the VQA task that requires understanding of both language and vision.

4.2.1 Text as Explanation

Hendricks et al. (2016) developed a model that generates text to justify its visual predictions. Their model focuses on the discriminating properties of the object in the image, jointly predicts a class label and also explains why the predicted label is appropriate for the image. Instead of looking up the definition or description of the detected object and using that as an explanation, the model uses the descriptive properties that are visible in the image and thus makes the reasoning more

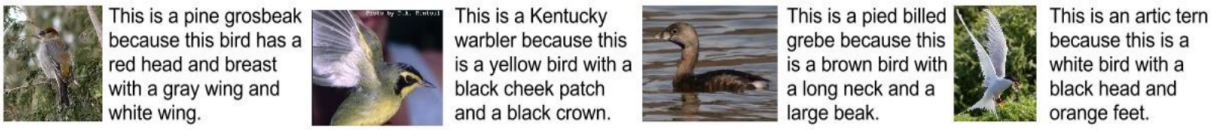


Figure 14: Sample text explanations generated by (Hendricks et al., 2016).

reliable. The authors show results on a fine-grained bird species classification dataset (Welinder et al., 2010). Figure 14 shows a sample of few images and explanations given by the authors on the aforementioned dataset.

Such explanations can be exploited to learn which system is reliable based on its justification. The idea behind using explanations as feature is to trust agreement between systems when their explanations are also similar just like in the case of provenance. We plan on improving the bird species classification task using the explanation text on the same Caltech-UCSD birds dataset consisting of 200 different categories of birds and approximately 6000 images (Welinder et al., 2010). By using multiple component systems that generate justification, the idea is to design auxiliary features that could aid the stacker in classification. We plan on using the machine translation metrics such as BLEU (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005) for comparing similarity between system generated explanations as features. Another interesting feature that captures the semantics of the text is to use embeddings of the words in the justification for calculating similarity. A pre-trained Word2Vec model could be fine-tuned on either the bird species description/definition dataset. We hope that using the explanations as features would enable the stacker to learn to rely on systems whose justification is congruent with its classification. In this way we would obtain an ensemble that is not just good in performance but also generates a consensus explanation for the ensemble.

4.2.2 Images as Explanation

Visual Question Answering (VQA) has gained a lot of attention in the past year (Goyal et al., 2016; Gella et al., 2016; Das et al., 2016; Andreas et al., 2016; Agrawal et al., 2016). VQA is a dataset containing open-ended questions about images (Antol et al., 2015). These questions require an understanding of vision, language and commonsense knowledge to answer them correctly. This dataset was released with a goal to better understand how deep learning systems come to a decision on the object-detection as well as to develop human trust in such systems. Figure 15 shows some images and questions from the VQA task as well as the ground truth answer.

Goyal et al. (2016) in their paper analyzed what parts of the image the VQA model focuses on while answering the question. They conclude that deep learning models attend to relevant parts of the image while answering the question. The parts of images that the models focus on can be thought of as visual explanations for answering the question. Their findings motivated us with the idea of using these visual explanations as auxiliary features.

The model's attention or the part of image that the model focuses on can be visualized using a heat-map of the image. The idea is to trust the agreement between systems when they also agree on the heat-map explanation. The agreement across these heat-maps can be used as auxiliary features. KL-divergence is one such metric for capturing the differences between heat maps. Another interesting feature is a measure of correlation between the heat maps. The idea behind using the visual

Figure 15: Random sample of Visual Question Answering (VQA) dataset. Taken from Antol et al. (2015)

explanation as features is very similar to that of using text explanations, that is, it would enable the stacker to better predict an answer by learning to rely on systems that "look" at the right region of the image while generating an answer. Using multiple VQA component models, we hope to improve the performance on the visual question answering task by using the justification features. One of the goals of explainable AI is to produce more explainable models[8]. A step towards the goal is to have VQA systems also produce explanations along with answers. These explanations can in turn be used to improve performance on the VQA task by using them as features for learning to rely on systems that agree on the explanation along with the answer.

Even more recently, Selvaraju et al. (2016) introduced the idea called Gradient-weighted Class Activation Mapping (Grad-CAM) that uses the class specific gradient information to produce a localization map for regions in the image that can be used as visual explanation. They performed an AMT experiment to evaluate human trust on various deep learning models. Given predictions from multiple models along with their visual explanations, human subjects were asked to rate the reliability of those models. They found that even though for instances when two models predicted the same object, one model was clearly rated better than the other (VGG over AlexNet) simply based on their visual explanations. The human trustworthiness also correlated with the performance of these models on the VQA dataset. These findings further reinforce our hypothesis that visual explanations serve as a crucial predictor for a system's performance on a task. Exploiting these explanations could lead to a boost in performance as well as increased human trust on the final system.

[8] http://www.darpa.mil/program/explainable-artificial-intelligence

4.2.3 Ensembles for Visual Question Answering (VQA)

Visual Question Answering is a challenge task introduced in 2015[9]. A VQA system takes as input an image and a natural language question about the image and produces a natural language answer as the output. As discusses in Section 4.2.2, VQA thus involves both language and vision understanding. The task has both open ended questions and questions with multiple choices. All the participating teams used some sort of deep learning for the task. The top performing team in 2016 used multimodal compact bilinear pooling to combine multimodal features (Fukui et al., 2016).

We propose to ensemble some of the participating teams in the VQA challenge using stacking with visual explanations as auxiliary features. It is challenging to ensemble systems based on their outputs for answers that are open-ended. On the other hand, for questions with multiple choices, ensembling is straightforward. Based on the analysis of the task by Antol et al. (2015), approximately 90% of the expected open-ended answers are one-word long. Our work would focus on using our stacking approach for questions with multiple choices and such one-word answers.

[9] http://www.visualqa.org

5 Conclusion

In this proposal, we considered the general machine learning problem of combining outputs from multiple systems to improve accuracy by using ensembling. We introduced stacking with auxiliary features (SWAF), a novel approach to ensemble multiple diverse systems. The auxiliary features enable the system to *learn* to appropriately use provenance that captures the "where" of the output as well as other instance-level information to aid the optimal integration of multiple systems. We demonstrated that our approach is a general, effective approach by applying it on three very different, challenging AI problems, the Cold Start Slot Filling and the Tri-lingual Entity Discovery and Linking tasks in the NLP domain and the ImageNet object detection task in computer vision. We obtained very promising results on all three tasks, beating the best component systems as well as other baseline ensembling methods. We achieved a new state-of-the-art on the two KBP tasks and significant improvements over baselines on the detection task.

We observed that other state-of-the-art ensembling techniques such as the Bipartite Graph Based Consensus Maximization (BGCM) and Mixtures of Experts (ME) models are not robust across tasks. The BGCM is good at precision but performs miserable at recall while the ME fails with many component systems because of its assumption that the underlying systems are trained on different feature sub-spaces, which is not always the case. On analyzing the results obtained by SWAF, we find that it does better when the component systems differ widely on their outputs and have low confidences, i.e. the errors produced by the systems are de-correlated. This leads us to conclude that the gain in performance from SWAF comes from output decisions that are difficult to make without context; however, using auxiliary features enables fusion of additional relevant information, allowing the stacker to make the right decision.

The stacking approach relies on training data for supervised learning and thus we proposed a novel approach to combine ensembles of systems with training data and ensemble of systems without historical training data. We presented results of our approach on the two KBP tasks, showing that a stacking-based approach to ensembling both supervised and unsupervised systems is very promising. We found that adding the unsupervised ensemble along with the shared systems specifically increased recall substantially, highlighting the importance of using systems that do not have historical training data.

Finally we proposed some short and long term future extensions to our work. In the short term we focus on using more semantic instance-level features for the three tasks and to explore non-lexical features that are language independent for the KBP tasks. Our first long-term focus is on using explanations as auxiliary features. The explanations are either textual or visual and we proposed ways on using them as features. Secondly, we plan on demonstrating our ensembling algorithm on the VQA task that requires understanding of both language and vision.

References

Aishwarya Agrawal, Dhruv Batra, and Devi Parikh. 2016. Analyzing the behavior of visual question answering models. *arXiv preprint arXiv:1606.07356*.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Learning to compose neural networks for question answering. *arXiv preprint arXiv:1601.01705*.

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 *TAC2013*.

Gabor Angeli, Victor Zhong, Danqi Chen, Arun Chaganty, Jason Bolton, Melvin Johnson Premkumar, Panupong Pasupat, Sonal Gupta, and Christopher D. Manning. 2015. Bootstrapped self training for knowledge base population. In *Proceedings of the Eighth Text Analysis Conference (TAC2015)*.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *The IEEE International Conference on Computer Vision (ICCV)*, December.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, volume 29, pages 65–72.

Michele Banko, Oren Etzioni, and Turing Center. 2008. The tradeoffs between open and traditional relation extraction. In *ACL08*, volume 8, pages 28–36.

Leo Breiman. 1996. Bagging predictors. *Machine learning*, 24(2):123–140.

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)*.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *EMNLP-CoNLL*, volume 7, pages 708–716.

Abhishek Das, Harsh Agrawal, C Lawrence Zitnick, Devi Parikh, and Dhruv Batra. 2016. Human Attention in Visual Question Answering: Do Humans and Deep Networks Look at the Same Regions? *arXiv preprint arXiv:1606.03556*.

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.

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.

David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. 2013. Learning factored representations in a deep mixture of experts. *arXiv preprint arXiv:1312.4314*.

Asif Ekbal and Sriparna Saha. 2013. Stacked ensemble coupled with feature selection for biomedical entity extraction. *Knowledge-Based Systems*, 46:22–32.

Joe Ellis, Jeremy Getman, Dana Fore, Neil Kuster, Zhiyi Song, Ann Bies, and Stephanie Strassel. 2015. Overview of linguistic resources for the TAC KBP 2015 evaluations: Methodologies and results. In *TAC 2015*.

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.

Nicolas Fauceglia, Yiu-Chang Lin, Xuezhe Ma, and Eduard Hovy. 2015. CMU System for Entity Discovery and Linking at TAC-KBP 2015. In *Proceedings of the Eighth Text Analysis Conference (TAC2015)*.

Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. 2010. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645.

Tim Finin, Dawn Lawrie, Paul McNamee, James Mayfield, Douglas Oard, Nanyun Peng, Ning Gao, Yiu-Chang Lin, Josh MacLin, and Tim Dowd. 2015. HLTCOE participation in TAC KBP 2015: Cold Start and TEDL. In *Proceedings of the Eighth Text Analysis Conference (TAC 2015)*.

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. 2003. Named entity recognition through classifier combination. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 168–171. ACL.

Matthew Francis-Landau, Greg Durrett, and Dan Klein. 2016. Capturing semantic similarity for entity linking with convolutional neural networks. In *Proceedings of the North American Association for Computational Linguistics*, San Diego, California, USA, June. Association for Computational Linguistics.

Yoav Freund and Robert E Schapire. 1995. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer.

Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. 2016. Multimodal compact bilinear pooling for visual question answering and visual grounding. *arXiv preprint arXiv:1606.01847*.

Jing Gao, Feng Liang, Wei Fan, Yizhou Sun, and Jiawei Han. 2009. Graph-based consensus maximization among multiple supervised and unsupervised models. In *NIPS2009*, pages 585–593.

Spandana Gella, Mirella Lapata, and Frank Keller. 2016. Unsupervised Visual Sense Disambiguation for Verbs using Multimodal Embeddings. *arXiv preprint arXiv:1603.09188*.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR2014*, June.

Ross Girshick. 2015. Fast R-CNN. In *International Conference on Computer Vision (ICCV2015)*.

Yash Goyal, Akrit Mohapatra, Devi Parikh, and Dhruv Batra. 2016. Towards Transparent AI Systems: Interpreting Visual Question Answering Models. *arXiv preprint arXiv:1608.08974*.

Zhengyan He, Shujie Liu, Yang Song, Mu Li, Ming Zhou, and Houfeng Wang. 2013. Efficient collective entity linking with stacking. In *EMNLP2013*, pages 426–435.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. *arXiv preprint arXiv:1512.03385*.

Benjamin Heinzerling, Alex Judea, and Michael Strube. 2015. HITS at TAC KBP 2015: Entity Discovery and Linking,and Event Nugget Detection. In *Proceedings of the Eighth Text Analysis Conference (TAC2015)*.

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 (EMNLP99)*, pages 187–194, College Park, MD.

Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. 2016. Generating Visual Explanations. *arXiv preprint arXiv:1603.08507*.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Dinesh Jayaraman and Kristen Grauman. 2014. Zero-shot recognition with unreliable attributes. In *Advances in Neural Information Processing Systems*, pages 3464–3472.

Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1148–1158. Association for Computational Linguistics.

Heng Ji, Joel Nothman, Ben Hachey, and Radu Florian. 2015. Overview of TAC-KBP2015 Tri-lingual Entity Discovery and Linking. In *Proceedings of the Eighth Text Analysis Conference (TAC2015)*.

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. *arXiv preprint arXiv:1408.5093*.

Bryan Kisiel, Bill McDowell, Matt Gardner, Ndapandula Nakashole, Emmanouil A. Platanios, Abulhair Saparov, Shashank Srivastava, Derry Wijaya, and Tom Mitchell. 2015. CMUML system for KBP 2015 Cold Start Slot Filling. In *Proceedings of the Eighth Text Analysis Conference (TAC2015)*.

Marcin Korytkowski, Leszek Rutkowski, and Rafał Scherer. 2016. Fast image classification by boosting fuzzy classifiers. *Information Sciences*, 327:175–182.

Andy Liaw and Matthew Wiener. 2002. Classification and regression by randomforest. *R news*, 2(3):18–22.

David G Lowe. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

Tomasz Malisiewicz, Abhinav Gupta, and Alexei A Efros. 2011. Ensemble of exemplar-svms for object detection and beyond. In *2011 International Conference on Computer Vision*, pages 89–96. IEEE.

David McClosky, Sebastian Riedel, Mihai Surdeanu, Andrew McCallum, and Christopher D Manning. 2012. Combining joint models for biomedical event extraction. *BMC Bioinformatics*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. pages 1003–1011. Association for Computational Linguistics (ACL2009).

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. 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 Third Conference on Natural language learning (NAACL2000)*, pages 63–69.

Lorenzo Peppoloni, Massimo Satler, Emanuel Luchetti, Carlo Alberto Avizzano, and Paolo Tripicchio. 2014. Stacked generalization for scene analysis and object recognition. In *Intelligent Engineering Systems (INES), 2014 18th International Conference on*, pages 215–220. IEEE.

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.

Nazneen Fatema Rajani and Raymond J. Mooney. 2016a. Stacking With Auxiliary Features. *ArXiv preprint arXiv:1605.08764*.

Nazneen Fatema Rajani and Raymond J. Mooney. 2016b. Combining Supervised and Unsupervised Ensembles for Knowledge Base Population. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP-16)*.

Nazneen Fatema Rajani, Vidhoon Viswanathan, Yinon Bentor, and Raymond J. Mooney. 2015. Stacked Ensembles of Information Extractors for Knowledge-Base Population. In *Association for Computational Linguistics (ACL2015)*, pages 177–187, Beijing, China, July.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS2015)*.

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.

Sebastian Riedel, David McClosky, Mihai Surdeanu, Andrew McCallum, and Christopher D Manning. 2011. Model combination for event extraction in bionlp 2011. In *Proceedings of the BioNLP Shared Task 2011 Workshop*, pages 51–55. ACL2011.

Miguel Rodriguez, Sean Goldberg, and Daisy Zhe Wang. 2015. University of Florida DSR lab system for KBP slot filler validation 2015. In *Proceedings of the Eighth Text Analysis Conference (TAC2015)*.

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)*.

Benjamin Roth, Nicholas Monath, David Belanger, Emma Strubell, Patrick Verga, and Andrew McCallum. 2015. Building knowledge bases with universal schema: Cold start and slot-filling approaches. In *Proceedings of the Eighth Text Analysis Conference (TAC2015)*.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252.

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)*.

Mark Sammons, Haoruo Peng, Yangqiu Song, Shyam Upadhyay, Chen-Tse Tsai, Pavankumar Reddy, Subhro Roy, and Dan Roth. 2015. Illinois CCG TAC 2015 Event Nugget, Entity Discovery and Linking, and Slot Filler Validation Systems. In *Proceedings of the Eighth Text Analysis Conference (TAC2015)*.

Ramprasaath R Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. 2016. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *arXiv preprint arXiv:1610.02391*.

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.

Avirup Sil and Radu Florian. 2016. One for All: Towards Language Independent Named Entity Linking.

Avirup Sil, Georgiana Dinu, and Radu Florian. 2015. The IBM systems for trilingual entity discovery and linking at TAC 2015. In *Proceedings of the Eighth Text Analysis Conference (TAC2015)*.

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

Stephen Soderland, Natalie Hawkins, Gene L. Kim, and Daniel S. Weld. 2015. University of Washington system for 2015 KBP cold start slot filling. In *Proceedings of the Eighth Text Analysis Conference (TAC2015)*.

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 (TAC2013)*.

Jasper RR Uijlings, Koen EA Van de Sande, Theo Gevers, and Arnold WM Smeulders. 2013. Selective search for object recognition. *International Journal of Computer Vision (IJCV)*, 104(2):154–171.

I-Jeng Wang, Edwina Liu, Cash Costello, and Christine Piatko. 2013. JHUAPL TAC-KBP2013 slot filler validation system. In *TAC2013*.

P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. 2010. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology.

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, 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)*.

Xiangzeng Zhou, Lei Xie, Peng Zhang, and Yanning Zhang. 2014. An ensemble of deep neural networks for object tracking. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 843–847. IEEE.