Stacking With Auxiliary Features

Nazneen Fatema Rajani
Department of Computer Science
University of Texas at Austin
nrajani@cs.utexas.edu

Raymond J. Mooney
Department of Computer Science
University of Texas at Austin
mooney@cs.utexas.edu

Abstract

Ensembling methods are well known for improving prediction accuracy. However, they are limited in the sense that they cannot effectively discriminate among component models. In this paper, we propose stacking with auxiliary features that learns to fuse additional relevant information from multiple component systems as well as input instances to improve performance. We use two types of auxiliary features – instance features and provenance features. The instance features enable the stacker to discriminate across input instances and 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 properties of the input instance. We demonstrate the success of our approach on three very different and challenging natural language and vision problems: Slot Filling, Entity Discovery and Linking, and ImageNet Object Detection. We obtain 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.

1 Introduction

Ensembling multiple systems is a well known standard approach to improving accuracy in machine learning [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. 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). 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.

In this paper, we consider the general machine learning problem of combining structured 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 *instance features*, and those that enable the stacker to discriminate across component systems, which we call *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. The exact form of provenance is specific to the task (e.g. a region in a text or an image) and captures "where" the system got its answer. Provenance indicates the origin of the generated output for each system and can be used to measure reliability of system outputs. Figure 1 gives a generalized overview of our approach to combining multiple system outputs. The idea behind using auxiliary features is that an output is more reliable if not only multiple systems produce it, but if they also agree on its provenance, and there are sufficient supporting instance features. The SWAF algorithm requires identifying appropriate instance and provenance features for a given task.

We use SWAF to demonstrate new state-of-the-art results on two of the three tasks and significant improvements over the best component system for the third task. The first two are in NLP and part of the NIST Knowledge Base Population (KBP) challenge – *Cold Start Slot-Filling* (CSSF)[†] and *Entity Discovery and Linking* (EDL) [Ji *et al.*, 2016]. The third task is in computer vision and part of the ImageNet 2015 challenge [Russakovsky *et al.*, 2015] – *Object Detection* from images. All the three tasks are difficult and well-known challenge problems. Auxiliary features for each of the tasks are identified based on the goal and supplementary resources for the problem. On all three tasks, SWAF outperforms the individual component systems as well as other ensembling methods such as stacking *without* auxiliary features, Mixtures of Experts [Jacobs *et al.*, 1991], and simple voting; verifying the generality and power of stacking with auxiliary features.

2 Background and Related Work

NIST annually conducts the Cold Start Slot Filling (CSSF) and Entity Discovery and Linking (EDL) competitions in the KBP track of the Text Analysis Conference (TAC). The goal of the 2016 CSSF task is to fill specific slots of information

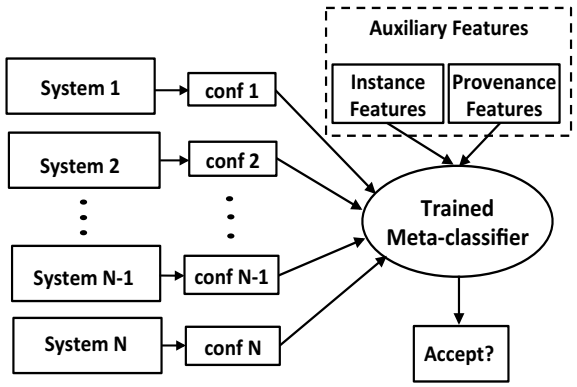

Figure 1: Our stacking approach to combining system outputs using confidence scores and two types of auxiliary features for improving prediction

for a given set of query entities based on a supplied text corpus. The goal of the 2016 EDL task [Ji *et al.*, 2016] is to discover entities based on a supplied text corpus as well as link these entities to an existing English Knowledge Base (KB) or cluster the mention with a NIL ID if the system could not successfully link the mention to any entity in the KB. The EDL task involved two foreign languages — Spanish and Chinese — along with English. The ImageNet object detection task [Russakovsky *et al.*, 2015] is a widely known annual challenge for evaluating vision systems on a large real world corpus. The objective of the task is to produce a list of object categories present in the image along with an axis-aligned bounding box indicating the position and scale of every instance of each detected object category.

For the CSSF task, participating teams employed a variety of techniques such as distant supervision, universal schemas, relevant document extraction, relation-modeling, open-IE, and inference [Ferguson *et al.*, 2016; Kisiel *et al.*, 2015]. The top performing 2016 CSSF system [Zhang *et al.*, 2016] leverages both distant supervision [Mintz *et al.*, 2009] and pattern-based relation extraction. Another system, *UMass_IESL* [Chang *et al.*, 2016], uses distant supervision, rule-based extractors, and semi-supervised matrix embedding methods. The EDL task on the other hand involves two NLP subtasks – Named Entity Recognition (NER) as well as disambiguation. The top performing 2016 EDL 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.*, 2016].

For ImageNet 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.

[†] http://www.nist.gov/tac/2016/KBP/ColdStart/index.html

Meta-learning addresses the question of how can we improve the performance of learning algorithms by using metadata about learning [Vilalta and Drissi, 2002]. Stacking is a type of meta-learning in which a meta-classifier is learned to combine the outputs of multiple underlying systems. The stacker learns a classification boundary based on the confidence scores provided by individual systems for each possible output. However, many times the scores produced by systems are not probabilities or not well calibrated and cannot be meaningfully compared. In such circumstances, it is beneficial to also have other reliable auxiliary features, as in our approach. Previously, it has been shown that stacking improves performance on slot filling [Viswanathan *et al.*, 2015]. However, our auxiliary features outperform this approach, resulting in a new state-of-the-art result on slot filling. To the best of our knowledge, there has been no prior work on ensembling for EDL, and our approach beats the current state-of-the-art system. 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. One past work combined basic ML models trained on the *same* set of features using stacking and obtained performance comparable to selecting the best classifier from the ensemble by cross validation on simple binary classification tasks [Džeroski and Ženko, 2004]. On the other hand, we use complex end-to-end diverse systems and evaluate on more difficult tasks with structured outputs.

Mixtures of Experts (ME) [Jacobs *et al.*, 1991] is another meta-learning algorithm in which the problem space is partitioned stochastically into a number of sub-spaces, with the idea that each learner (expert) is specialized to a particular subspace. ME uses divide-and-conquer to soft switch between learners covering different sub-spaces of the input by employing a supervised gating network that is learned using Expectation-Maximization (EM). More recently, the ME algorithm was extended to use a different gating network at each layer in a multilayer network, forming a Deep Mixture of Experts [Eigen *et al.*, 2013]. We compare our results to ME because of its similarity to the use of instance features in SWAF by adapting the ensemble decision based on properties of the specific instance being classified.

3 Overview of the Tasks

In this section we give a short overview of each of the three tasks we used to evaluate SWAF.

3.1 Cold Start Slot Filling

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 evaluation included a total of forty-one fixed slots.[‡] Some slots (like per:age) are *single-valued* while others (like per:children) are *list-valued* i.e., they can take multiple slot fillers.

[‡] https://tac.nist.gov/2016/KBP/ColdStart/guidelines/TAC_KBP_2016_ColdStartTaskDescription_1.0.pdf

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 and newswire, each identified by a unique ID. The output is a set of slot fills for each 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 optionally provide a confidence score to indicate their certainty in the extracted information.

3.2 Entity Discovery and Linking

The goal of EDL 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 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. A version of FreeBase [Bollacker *et al.*, 2008] is used as the KB.

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.

3.3 Object Detection for Images

The goal of the 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 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.

4 SWAF Method

Figure 1 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 sections below.

4.1 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 weigh 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 that performs better than each individual component system. In our approach, structured outputs must be represented as *key-value* pairs. The meta-classifier makes a binary decision whether or not to accept each distinct output pair. Thus before deploying the algorithm on a task, it is crucial to identify the *key* which serves as a unique handle for ensembling 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 ensemble 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 are labeled as "correct" by the meta-classifier are kept while the others are removed.

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 EDL, we define the key to be the *KB ID (or NIL)* 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 use 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.

For the two NLP tasks, we used an L1 regularized linear SVM weighted by the number of instances for each class, as a meta-classifier for stacking. For the object detection task, we used an SVM with an RBF kernel. A small random sample of the training set (10%) was used as validation data to set the hyper-parameters for each of the tasks.

4.2 Auxiliary Features

As discussed earlier, we use two types of auxiliary features: *provenance* and *instance*. For provenance features, we use the provenance information for each generated output pair. Provenance indicates "where" the system found the 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, it is a bounding box in the image.

For CSSF, if a system successfully extracts a relation then it must provide provenance indicating the location of the extracted slot fill in the corpus. For EDL, 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 de-

tects a target category then it must provide a bounding box localizing the object in the image. The bounding box is in the form $\langle x_{min}, y_{min}, x_{max}, y_{max} \rangle$. 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.

The idea behind using provenance information in auxiliary features is that an output is more reliable if multiple systems agree not only on the decision itself, but also on its provenance. In order to enable the stacker to leverage this information, we develop features that measure provenance agreement *across* systems. The Jaccard similarity coefficient is a statistical measure of similarity between sets and is thus useful in measuring the degree of overlap between the provenance provided by systems. For the CSSF and EDL tasks, the provenance strings are used to capture similarity as follows. For a given *key*, if N systems that generate a *value* have the same *docid* for their document provenance, then the provenance overlap (PO) score for a system n is calculated as the intersection of the provenance strings divided by their union, averaged over all other systems:

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

Thus, systems that generate a *value* from *different* documents for the same *key* have zero provenance overlap. For the object detection task, the Jaccard coefficient is used to measure the overlap between bounding boxes across systems. 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 EDL 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. We note that the use of provenance as features does not require access to the large corpus of documents or images and is thus computationally inexpensive.

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). For example, a classifier could learn not to trust an object detection output for the 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 a one-hot encoding of the slot type (e.g. per:age) and for the EDL task, we use a one-hot encoding of the entity type (PER/ORG/GPE/FAC/LOC) as instance features. Another instance 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 EDL 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 EDL systems provide as provenance is the *value* document for both the tasks. 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 the *key* and *value* documents should be similar because they are discussing the same entity.

Recently, [Francis-Landau *et al.*, 2016] used contextual information to disambiguate entity mentions for the entity-linking problem and showed that measuring semantic similarity at different granularities of the mention text and potential KB document leads to improved performance. We consider the mention text at three levels of granularity – title, mention sentence and entire mention document. The potential KB document is split into two – title and entire document. We first embed words into vectors by training a word2vec model [Mikolov *et al.*, 2013] with word vector dimension of 100, window-size set to 8, 10 negative samples and 10 iterations on FreeBase. Thereafter, we use these vectors to measure semantic similarity between words at each granularity of the mention text and the potential KB document. In this way, we obtain six pairwise similarity measures which are also used as instance features for the EDL task.

For the ImageNet task, we use two types of instance features. First, we use a one-hot encoding of the object category (total 200). Second, we use a bag of visual words for the image using Scale-Invariant Feature Transform (SIFT) as the feature descriptor [Lowe, 2004] as well as the $4,096$ features from VGGNet's $fc7$ layer [Simonyan and Zisserman, 2015]. Note that some underlying object detection systems also use these features for classification and we show that using them for learning the top-level meta-classifier further boosts the performance.

4.3 Post-processing

Once we obtain the decision for each key-value pair from the stacker, we perform some final post-processing to produce output that is in the same format as that generated by an individual 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 only include the fill with the highest meta-classifier confidence. For the EDL task, for each entity mention link that is classified as correct, if the link is a KB ID then we include it in the final output, but if the link is a NIL ID then we keep it aside until all mention links are

Method	Precision	Recall	F1
Stacking with both provenance + instance auxiliary features	0.258	**0.439**	**0.324**
Stacking with just provenance auxiliary features	0.252	0.377	0.302
Stacking with just instance auxiliary features	0.257	0.346	0.295
Stacking without auxiliary features	**0.311**	0.253	0.279
Top ranked CSSF system in 2016 [Zhang *et al.*, 2016]	0.265	0.302	0.260
Oracle Voting baseline (4 or more systems must agree)	0.191	0.379	0.206
Mixtures of Experts (ME) model	0.168	0.321	0.180

Table 1: Results on 2016 Cold Start Slot Filling (CSSF) task using the official NIST scorer

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 ImageNet object detection, 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 overlap with other systems, and thus is used as the bounding box for the ensemble. When there are only two systems that produced an ouput, this method does not discriminate so we use the bounding box produced by the system with the higher confidence score. We also experimented with using the *union* or *intersection* of bounding boxes across systems as the aggregate bounding box for the ensemble, but this approach is heavily penalized by the ImageNet evaluation metric. For a standard sized image (larger than 25×25 pixels), the detection scorer considers an object to be localized correctly iff the IOU of the output bounding box and the ground-truth is ≥ 0.5.

5 Experimental Results

We evaluated stacking with auxiliary features (SWAF) on three challenge problems, comparing our full system to various ablations and prior results. All KBP results were obtained using the official NIST scorer provided after the competition ended.[§] For object detection, we used the scorer provided with the ImageNet devkit.

SWAF relies on training data for learning and thus for the two KBP tasks, we only use systems that participated in both the 2015 and 2016 competitions. This allows us to train the stacker on 2015 system outputs and use the trained model to evaluate on the 2016 data. In this way, we used 8 common systems for CSSF and 6 for EDL. We were unable to obtain all 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 [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.

For the CSSF task, systems are evaluated against a gold standard using precision, recall and F1 scores for the extracted slot fills. The EDL evaluation uses the mention CEAF metric [Ji *et 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 EDL tasks in 2016. For the 2015 object detection task, we were unable to obtain 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 4 for both the CSSF and EDL tasks and 1 for the object detection task. We note that oracle voting is "cheating" to give the best possible voting baseline.

Tables 1 and 2 show the results for CSSF and EDL, respectively. Stacking with both instance and provenance features consistently performed the best on both tasks, outperforming all ablations, the ME algorithm, as well as the top ranked systems from the 2016 competition. Oracle voting performs very poorly, indicating that naive ensembling is not advantageous. The relative ranking of our approaches is the same on both the CSSF and EDL tasks, thus demonstrating that our

[§] http://www.nist.gov/tac/2016/KBP/ColdStart/tools.html, https://github.com/wikilinks/neleval

Method	Precision	Recall	F1
Stacking with both provenance + instance auxiliary features	0.739	**0.600**	**0.662**
Stacking with just provenance auxiliary features	**0.767**	0.544	0.637
Stacking with just instance auxiliary features	0.752	0.542	0.630
Stacking without auxiliary features	0.723	0.537	0.616
Top ranked EDL system in 2016 [Sil *et al.*, 2016]	0.717	0.517	0.601
Mixtures of Experts (ME) model	0.721	0.494	0.587
Oracle Voting baseline (4 or more systems must agree)	0.588	0.412	0.485

Table 2: Results on 2016 Entity Discovery and Linking (EDL) task using the official NIST scorer and the CEAFm metric

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

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 since its performance is inversely proportional to the number of component systems. Since the CSSF task had more systems (i.e. 8) than the EDL 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. By training on an increasing percentage of the previous year's data and retesting on 2016 data at each step, we concluded that although systems get better over the years, it is still advantageous to train on a previous year's output.

Table 3 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 as part of our ensemble, we would expect to obtain a new state-of-the-art result. Since we use cross-validation for obtaining these results, we performed a pairwise t-test with significance level 0.05 and found that using any ablation of stacking with auxiliary features is significantly better (p-value< 0.05) than using the best component system, although using stacking alone is not significantly worse. The ME algorithm performs significantly better than the individual components since it works well given the small number of component systems (i.e. 3), but it is still worse than our best approach. 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. An analysis of the results showed that SWAF particularly does well on localizing objects in images that have multiple instances of the *same* object, i.e. the image could be considered to be "cluttered".

6 Conclusion

In this paper, we introduced stacking with auxiliary features (SWAF), a novel approach to ensemble multiple diverse system outputs. The auxiliary features enable the system to *learn* to appropriately use provenance and instance information to aid the optimal integration of multiple systems. We demonstrated that our approach is a general, effective approach by applying it to three very different, challenging AI problems requiring structured output: Cold Start Slot Filling and the Entity Discovery and Linking tasks in NLP, and ImageNet object detection in computer vision. SWAF obtained very promising results on all three tasks, outperforming the best component systems as well as other ensembling methods. The approach provides an overall F1 score of 32.4% on the 2016 KBP CSSF task and CEAFm F1 of 66.2% on the 2016 KBP EDL, and an overall mAP of 50.6% on the ImageNet object detection task. We achieve a new state-of-the-art on the two KBP tasks and significant improvements over baselines on the detection task.

We observe that the well known mixture-of-experts method is not robust 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 outputs differ widely and have low confidences. 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.

Acknowledgement

This research was supported by the DARPA DEFT program under AFRL grant FA8750-13-2-0026.

References

[Bollacker *et al.*, 2008] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, 2008.

[Chang *et al.*, 2016] H. Chang, M. Abdurrahman, A. Liu, J. Tian-Zheng Wei, A. Traylor, A. Nagesh, N. Monath, P. Verga, E. Strubell, and A. McCallum. Extracting Multilingual Relations under Limited Resources: TAC 2016 Cold-Start KB construction and Slot-Filling using Compositional Universal Schema. In *Proceedings of TAC*, 2016.

[Dietterich, 2000] T. Dietterich. Ensemble methods in machine learning. In *First International Workshop on Multiple Classifier Systems, Lecture Notes in Computer Science*, pages 1–15. Springer-Verlag, 2000.

[Džeroski and Ženko, 2004] Saso Džeroski and Bernard Ženko. Is combining classifiers with stacking better than selecting the best one? *Machine learning*, 54(3):255–273, 2004.

[Eigen *et al.*, 2013] D. Eigen, M. Ranzato, and I. Sutskever. Learning factored representations in a deep mixture of experts. *arXiv preprint arXiv:1312.4314*, 2013.

[Felzenszwalb *et al.*, 2010] P. Felzenszwalb, D. Girshick, R.and McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[Ferguson *et al.*, 2016] J. Ferguson, C. Lockard, N. Hawkins, S. Soderland, H. Hajishirzi, and D. Weld. University of washington tac-kbp 2016 system description. In *Proceedings of TAC*, 2016.

[Francis-Landau *et al.*, 2016] M. Francis-Landau, G. Durrett, and D. Klein. Capturing semantic similarity for entity linking with convolutional neural networks. In *Proceedings of the NAACL*, 2016.

[Girshick, 2015] Ross Girshick. Fast R-CNN. In *International Conference on Computer Vision*, 2015.

[He *et al.*, 2015] K. He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[Jacobs *et al.*, 1991] R.A Jacobs, M. Jordan, S. Nowlan, and G. Hinton. Adaptive mixtures of local experts. *Neural computation*, 1991.

[Ji *et al.*, 2015] H. Ji, J. Nothman, B. Hachey, and R. Florian. Overview of TAC-KBP2015 Tri-lingual Entity Discovery and Linking. In *Proceedings of TAC*, 2015.

[Ji *et al.*, 2016] H. Ji, J. Nothman, and H. Trang Dang. Overview of TAC-KBP2016 Tri-lingual EDL and its impact on end-to-end Cold-Start KBP. In *Proceedings of TAC*, 2016.

[Jia *et al.*, 2014] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[Kisiel *et al.*, 2015] B. Kisiel, B. McDowell, M. Gardner, N. Nakashole, E. Platanios, A. Saparov, S. Srivastava, D. Wijaya, and T. Mitchell. CMUML system for KBP 2015 Cold Start Slot Filling. In *Proceedings of the Eighth TAC*, 2015.

[Lowe, 2004] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004.

[Mikolov *et al.*, 2013] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, 2013.

[Mintz *et al.*, 2009] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. pages 1003–1011. ACL, 2009.

[Peppoloni *et al.*, 2014] L. Peppoloni, M. Satler, E. Luchetti, C. Avizzano, and P. Tripicchio. Stacked generalization for scene analysis and object recognition. In *Proceedings of INES*, 2014.

[Ren *et al.*, 2015] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems*, 2015.

[Russakovsky *et al.*, 2015] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 115(3), 2015.

[Sil *et al.*, 2016] A. Sil, G. Dinu, and R. Florian. The IBM systems for trilingual entity discovery and linking at TAC 2016. In *Proceedings of TAC*, 2016.

[Sill *et al.*, 2009] J. Sill, G. Takács, L. Mackey, and D. Lin. Feature-weighted linear stacking. *arXiv preprint arXiv:0911.0460*, 2009.

[Simonyan and Zisserman, 2015] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-scale Image Recognition. In *Proceedings of ICLR*, 2015.

[Uijlings *et al.*, 2013] J. Uijlings, K. Van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. *IJCV*, 2013.

[Vilalta and Drissi, 2002] R. Vilalta and Y. Drissi. A Perspective View and Survey of Meta-learning. *Artificial Intelligence Review*, 2002.

[Viswanathan *et al.*, 2015] V. Viswanathan, N. Rajani, Y. Bentor, and R. Mooney. Stacked ensembles of information extractors for knowledge-base population. In *Proceedings of ACL*, 2015.

[Wolpert, 1992] D. Wolpert. Stacked generalization. *Neural Networks*, 5, 1992.

[Zhang *et al.*, 2016] Y. Zhang, A. Chaganty, A. Paranjape, D. Chen, J. Bolton, P. Qi, and C. Manning. Stanford at TAC KBP 2016: Sealing Pipeline Leaks and Understanding Chinese. In *Proceedings of TAC*, 2016.