Evaluating Unsupervised Ensembles when applied to Word Sense Induction

Keith Stevens[1,2]
[1]University of California Los Angeles; Los Angeles , California, USA
[2]Lawrence Livermore National Lab; Livermore, California, USA*
kstevens@cs.ucla.edu

Abstract

Ensembles combine knowledge from distinct machine learning approaches into a general flexible system. While supervised ensembles frequently show great benefit, unsupervised ensembles prove to be more challenging. We propose evaluating various unsupervised ensembles when applied to the unsupervised task of Word Sense Induction with a framework for combining diverse feature spaces and clustering algorithms. We evaluate our system using standard shared tasks and also introduce new automated semantic evaluations and supervised baselines, both of which highlight the current limitations of existing Word Sense Induction evaluations.

1 Introduction

Machine learning problems often benefit from many differing solutions using ensembles (Dietterich, 2000) and supervised Natural Language Processing tasks have been no exception. However, use of unsupervised ensembles in NLP tasks has not yet been rigorously evaluated. Brody et al. (2006) first considered unsupervised ensembles by combining four state of the art Word Sense Disambiguation systems using a simple voting scheme with much success. Later, Brody and Lapata (2009) combined different feature sets using a probabilistic Word Sense Induction model and found that only some combinations produced an improved system. These early and limited evaluations show both the promise and drawback of combining different unsupervised models: particular combinations provide a benefit but selecting these combinations is non-trivial.

We propose applying a new and more general framework for combining unsupervised systems known as Ensemble Clustering to unsupervised NLP systems and focus on the fully unsupervised task of Word Sense Induction. Ensemble Clustering can combine together multiple and diverse clustering algorithms or feature spaces and has been shown to noticeably improve clustering accuracy for both text based datasets and other datasets (Monti et al., 2003; Strehl et al., 2002). Since Word Sense Induction is fundamentally a clustering problem, with many variations, it serves well as a NLP case study for Ensemble Clustering.

The task of Word Sense Induction extends the problem of Word Sense Disambiguation by simply assuming that a model must first learn and define a sense inventory before disambiguating multi-sense words. This induction step frees the disambiguation process from any fixed sense inventory and can instead flexibly define senses based on observed patterns within a dataset (Pedersen, 2006). However, this induction step has proven to be greatly challenging, in the most recent shared tasks, induction systems either appear to perform poorly or fail to outperform the simple Most Frequent Sense baseline (Agirre and Soroa, 2007a; Manandhar et al., 2010).

In this work, we propose applying Ensemble Clustering as a general framework for combining not only different feature spaces but also a variety of different clustering algorithms. Within this framework we will explore which types of models should be combined and how to best combine them. In addition, we propose two new evaluations: (1) new semantic coherence measures that evaluate the seman-

*This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 (LLNL-CONF-530791).

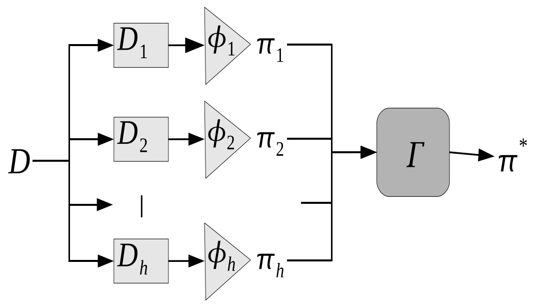

Figure 1: The Ensemble Clustering model: individual clustering algorithms partition perturbations of the dataset and all partitions are combined via a *consensus function* to create a final solution, π^*.

tic quality and uniqueness of induced word senses without referring to an external sense inventory (2) and a new set of baseline systems based on supervised learning algorithms. With the new evaluations and a framework for combining general induction models, we intend to find not only improved models but a better understanding of how to improve later induction models.

2 Consensus Clustering

Ensemble Clustering presents a new method for combining together arbitrary clustering algorithms without any supervision (Monti et al., 2003; Strehl et al., 2002). The method adapts simple boosting and voting approaches from supervised ensembles to merge together diverse clustering partitions into a single consensus solution. Ensemble Clustering forms a single consensus partition by processing a data set in two steps: (1) create a diverse set of ensembles that each partition some perturbation of the full dataset and (2) find the median partition that best agrees with each ensemble's partition. Figure 1 visually displays these two steps.

Variation in these two steps accounts for the wide variety of Ensemble Clustering approaches. Each ensemble can be created from either a large collection of distinct clustering algorithms or through a boosting approach where the same algorithm is trained on variations of the dataset. Finding the median partition turns out to be an NP-Complete problem under most settings (Goder and Filkov, 2008) and thus must be approximated with one of several heuristics. We consider several well tested approaches to both steps.

Formally, we define Ensemble Clustering to operate over a dataset of N elements: $D = \{d_1, \dots, d_N\}$. Ensemble Clustering then creates H ensembles that each partition a perturbation D_h of D to create H partitions, $\Pi = \{\pi_1, \cdots, \pi_H\}$. The consensus algorithm then approximates the best consensus partition π^* that satisfies:

$$\underset{\pi^*}{\operatorname{argmin}} \sum_{\pi_h \in \Pi} d(\pi_h, \pi^*) \tag{1}$$

according to some distance metric $d(\pi_i, \pi_j)$ between two partitions. We use the *symmetric difference distance* as $d(\pi_i, \pi_j)$. Let P_i be the set of co-cluster data points in π_i. The distance metric is then defined to be

$$d(\pi_1, \pi_2) = |P_1 \setminus P_2| + |P_2 \setminus P_1|$$

2.1 Forming Ensembles

Ensemble clustering can combine together overlapping decisions from many different clustering algorithms or it can similarly boost the performance of a single algorithm by using different parameters. We consider two simple formulations of ensemble creation: *Homogeneous Ensembles* and *Heterogeneous Ensembles*. We secondly consider approaches for combining the two creation methods.

Homogeneous Ensembles partition randomly sampled subsets of the data points from D without replacement. By sampling without replacement, each ensemble will likely see different representations of each cluster and can specialize its partition the around observed subset. Furthermore, each ensemble will observe less noise and can better define each true cluster (Monti et al., 2003). We note that since each ensemble only observes an incomplete subset of D, some clusters may not be represented at all in some partitions.

Heterogeneous Ensembles create diverse partitions by simply using complete partitions over D from different clustering algorithms, either due to different parameters or due to completely different clustering models (Strehl et al., 2002).

Combined Heterogeneous and Homogeneous Ensembles can be created by creating many homogeneous variations of each distinct clustering algorithm within a heterogeneous ensemble. In this framework, each single method can be boosted by subsampling the data in order to observe the true clusters and then combined with other algorithms using differing cluttering criteria.

2.2 Combining data partitions

Given the set of partitions, $\Pi = \{\pi_i, \cdots, \pi_h\}$, the consensus algorithm must find a final partition, π^* that best minimizes Equation 1. We find an approximation to π^* using the following algorithms.

Agglomerative Clustering first creates a *consensus matrix*, $\mathcal{M}$ that records the aggregate decisions made by each partition. Formally, $\mathcal{M}$ records the fraction of partitions that observed two data point *and* assigned them to the same cluster:

$$\mathcal{M}(i,j) = \frac{\sum_{k=1}^{h} 1\{d_i, d_j \in \pi_k^c\}}{\sum_{k=1}^{h} 1\{d_i, d_j \in \pi_k\}}$$

Where d_i refers to element i, π_k^c refers to cluster c in partition π_k, and $1\{*\}$ is the indicator function. The consensus partition, π^* is then the result of creating C partitions with Agglomerative Clustering using the Average Link criterion and $\mathcal{M}$ as the similarity between each data point (Monti et al., 2003).

Best of K simply sets π^* as the partition $\pi_h \in \Pi$ that minimizes Equation 1 (Goder and Filkov, 2008).

Best One Element Move begins with an initial consensus partition $\hat{\pi}^*$ and repeatedly changes the assignment of a single data point such that Equation 1 is minimized and repeats until no move can be found. We initialize this with Best of K.

Filtered Stochastic Best One Element Move also begins with an initial consensus partition $\hat{\pi}^*$ and repeatedly finds the best one element move, but does not compare against every partition in Π for each iteration. It instead maintains a history of move costs and updates that history with a stochastically selected partition from Π for each move iteration and ends after some fixed number of iterations (Zheng et al., 2011).

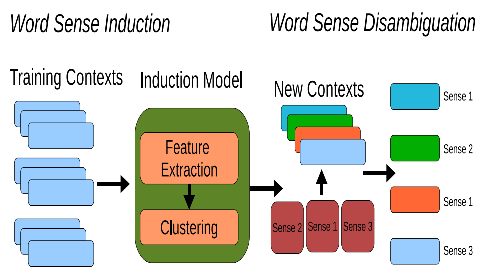

Figure 2: The general Word Sense Induction Model: models extract distributional data from contexts and induce senses by clustering the extracted information. Models then use representations of each sense to disambiguate new contexts.

3 Word Sense Induction Models

Word Sense Induction models define word senses in terms of the distributional hypothesis, whereby the meaning of a word can be defined by the surrounding context (Haris, 1985). Rather than form a single representation for any word, induction models represent the distinct contexts surrounding a multi-sense word and find commonalities between the observed contexts by clustering. These similar contexts then define a particular *word sense* and can be used to later recognize later instances of the sense, Figure 2.

Models can be roughly categorized based on their context model and their clustering algorithm into two categories: feature vector methods and graph methods. Feature vector methods simply transform each context into a feature vector that records contextual information and then cluster with any algorithm that can partition individual data points. Graph methods build a large distributional graph that models lexical features from all contexts and then partitions the graph using a graph-based clustering algorithm. In both cases, models disambiguate new uses of a word by finding the sense with the most features in common with the new context.

3.1 Context Models

Context models follow the distributional hypothesis by encoding various lexical and syntactic features that frequently occur with a multi-sense word. Each context model records different levels of information, and in different formats, but are limited to features available from syntactic parsing. Below we summarize our context models which are based on previous induction systems:

Word Co-occurence (WoC) acts as the core feature vector method and has been at the core of nearly all systems that model distributional semantics (Pedersen, 2006). The WoC model represents each context simply as the words within $\pm W$ words from the multi-sense word. Each co-occurring word is weighted by the number of times it occurs within the window.

Parts of Speech (PoS) extends the WoC model by appending each lexical feature with its part of speech. This provides a simple disambiguation of each feature so that words with multiple parts of speech are not conflated into the same feature. (Brody et al., 2006).

Dependency Relations (DR) restrains word co-occurrence to words that are reachable from the multi-sense word via a syntactic parse composed of dependency relationships limited by some length (Padó and Lapata, 2007). We treat each reachable word and the last relation in the path as a feature (Van de Cruys and Apidianaki, 2011).

Second Order Co-occurrence (SndOrd) provides a rough compositional approach to representing sentences that utilizes word co-occurrence and partially solves the data sparsity problem observed with the WoC model. The SndOrd model first builds a large distributional vector for each word in a corpus and then forms context vectors by adding the distributional vector for each co-occurring context word (Pedersen, 2006).

Graph models encode rich amounts of linguistic information for all contexts as a large distributional graph. Each co-occurring context word is assigned a node in the graph and edges are formed between any words that co-occur in the same context. The graph is refined by comparing nodes and edges to a large representative corpus and dropping some occurrences (Klapaftis and Manandhar, 2010).

Latent Factor Models projects co-occurrence information into a latent feature space that ties together relationships between otherwise distinct features. We consider three latent models: the Singular Value Decomposition (SVD) (Schütze, 1998), Non-negative Matrix Factorization (NMF) (Van de Cruys and Apidianaki, 2011), and Latent Dirichlet Allocation (Brody and Lapata, 2009). We note that SVD and NMF operate as a second step over any feature vector model whereas LDA is a standalone model.

3.2 Clustering Algorithms

Distributional clustering serves as the main tool for detecting distinct word senses. Each algorithm makes unique assumptions about the distribution of the dataset and should thus serve well as diverse models, as needed by supervised ensembles (Dietterich, 2000). While many WSI models automatically estimate the number of clusters for a word, we initially simplify our evaluation by assuming the number of clusters is known a priori and instead focus on the distinct underlying clustering algorithms. Below we briefly summarize each base algorithm:

K-Means operates over feature vectors and iteratively refines clusters by associating each context vector with its most representative centroid and then reformulating the centroid (Pedersen and Kulkarni, 2006).

Hierarchical Agglomerative Clustering can be applied to both feature vectors and collocation graphs. In both cases, each sentences or collocation vertex is placed in their own clusters and then the two most similar clusters are merged together into a new cluster (Schütze, 1998).

Spectral Clustering separates an associativity matrix by finding the cut with the lowest conductance. We consider two forms of spectral clustering: EigenCluster (Cheng et al., 2006), a method originally designed to cluster snippets for search results into semantically related categories, and GSpec (Ng et al., 2001), a method that directly clusters a collocation graph.

Random Graph Walks performs a series of random walks through a collocation graph in order to discover nodes that serve as central discriminative points in the graph and tightly connected components in the graph. We consider Chinese Whispers (Klapaftis and Manandhar, 2010) and a hub selection algorithm (Agirre and Soroa, 2007b).

4 Proposed Evaluation

We first propose evaluating ensemble configurations of Word Sense Induction models using the standard shared tasks from SemEval-1 (Agirre and Soroa, 2007a) and SemEval-2 (Manandhar et al., 2010). We then propose comparing these results, and past SemEval results, to supervised baselines as a gauge of how well the algorithms do compared to more informed models. We then finally propose an intrinsic evaluation that rates the semantic interpretability and uniqueness of each induced sense.

Evaluating Ensemble Configurations must be done to determine which variation of Ensemble Clustering best applies to the Word Sense Induction tasks. Preliminary research has shown that Homogeneous ensemble combined with the HAC consensus function typically improve base models while combining heterogeneous induction models *greatly* reduces performance. We thus propose various sets of ensembles to evaluate whether or not certain context models or clustering algorithms can be effectively combined:

1. mixing different feature vector models with the same clustering algorithm,
2. mixing different clustering algorithms using the same context model,
3. mixing feature vector context models and graph context models using matching clustering algorithms,
4. mixing all possible models,
5. and improving each heterogeneous algorithm by first boosting them with homogeneous ensembles.

SemEval Shared Tasks provide a shared corpus and evaluations for comparing different WSI Models. Both shared tasks from SemEval provide a corpus of training data for 100 multi-sense words and then compare the induced sense labels generated for a set of test contexts with human annotated sense using a fixed sense inventory. The task provides two evaluations: an *unsupervised* evaluation that treats each set of induced senses as a clustering solution and measures accuracy with simple metrics such as the Paired F-Score, V-Measure, and Adjusted Mutual Information; and a *supervised* evaluation that builds a simple supervised word sense disambiguation system using the sense labels (Agirre and Soroa, 2007a; Manandhar et al., 2010).

Supervised Baselines should set an upper limit on the performance we can expect from most unsupervised algorithms, as has been observed in other NLP tasks. We train these baselines by using feature vector models in combination with the SemEval-1 dataset[1]. We propose several standard supervised machine learning algorithms as different baselines: Naive Bayes, Logistic Regression, Decision Trees, Support Vector Machines, and various ensembles of each such as simple Bagged Ensembles.

Semantic Coherence evaluations balance the shared task evaluations by functioning without a sense inventory. Any evaluation against an existing inventory cannot accurately measure newly detected senses, overlapping senses, or different sense granularities. Therefore, our proposed *sense coherence* measures focus on the semantic quality of a sense, adapted from topic coherence measures (Newman et al., 2010; Mimno et al., 2011). These evaluate the degree to which features in an induced sense describe the meaning of the word sense, where highly related features constitute a more coherent sense and unrelated features indicate an incoherent sense. Furthermore, we adapt the coherence metric to evaluate the amount of semantic overlap between any two induced senses.

5 Concluding Remarks

This research will better establish the benefit of Ensemble Clustering when applied to unsupervised Natural Language Processing tasks that center around clustering by examining which feature spaces and algorithms can be effectively combined along with different different ensemble configurations. Furthermore, this work will create new baselines that evaluate the inherent challenge of Word Sense Induction and new automated and knowledge lean measurements that better evaluate new or overlapping senses learned by induction systems. All of the work will be provided as part of a flexible open source framework that can later be applied to new context models and clustering algorithms.

[1]We cannot use graph context models as they do not model contexts individually, nor can we use the SemEval-2 dataset because the training set lacks sense labels needed for training supervised systems

References

Eneko Agirre and Aitor Soroa. 2007a. Semeval-2007 task 02: evaluating word sense induction and discrimination systems. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, pages 7–12, Stroudsburg, PA, USA. Association for Computational Linguistics.

Eneko Agirre and Aitor Soroa. 2007b. Ubc-as: a graph based unsupervised system for induction and classification. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, pages 346–349, Stroudsburg, PA, USA. Association for Computational Linguistics.

Samuel Brody and Mirella Lapata. 2009. Bayesian word sense induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 103–111, Stroudsburg, PA, USA. Association for Computational Linguistics.

Samuel Brody, Roberto Navigli, and Mirella Lapata. 2006. Ensemble methods for unsupervised wsd. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 97–104, Sydney, Australia, July. Association for Computational Linguistics.

David Cheng, Ravi Kannan, Santosh Vempala, and Grant Wang. 2006. A divide-and-merge methodology for clustering. *ACM Trans. Database Syst.*, 31:1499–1525, December.

Thomas G. Dietterich. 2000. Ensemble methods in machine learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, MCS '00, pages 1–15, London, UK. Springer-Verlag.

Andrey Goder and Valdimir Filkov, 2008. *Consensus Clustering Algorithms: Comparison and Refinement.*, pages 109–117.

Zellig Haris, 1985. *Distributional Structure*, pages 26–47. Oxford University Press.

Ioannis P. Klapaftis and Suresh Manandhar. 2010. Word sense induction & disambiguation using hierarchical random graphs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 745–755, Stroudsburg, PA, USA. Association for Computational Linguistics.

Suresh Manandhar, Ioannis Klapaftis, Dmitriy Dligach, and Sameer Pradhan. 2010. Semeval-2010 task 14: Word sense induction & disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 63–68, Uppsala, Sweden, July. Association for Computational Linguistics.

David Mimno, Hanna Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. 2011. Optimizing semantic coherence in topic models. In *Proceedings of the 2011 Conference on Emperical Methods in Natural Language Processing*, pages 262–272, Edinburgh, Scotland, UK. Association of Computational Linguistics.

Stefano Monti, Pablo Tamayo, Jill Mesirov, and Todd Golub. 2003. Consensus clustering – a resampling-based method for class discovery and visualization of gene expression microarray data. *Machine Learning*, 52:91–118, July.

David Newman, Youn Noh, Edmund Talley, Sarvnaz Karimi, and Timothy Baldwin. 2010. Evaluating topic models for digital libraries. In *Proceedings of the 10th annual joint conference on Digital libraries*, JCDL '10, pages 215–224, New York, NY, USA. ACM.

A. Ng, M. Jordan, and Y. Weiss. 2001. On Spectral Clustering: Analysis and an algorithm. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, pages 849–856. MIT Press.

Sebastian Padó and Mirella Lapata. 2007. Dependency-Based Construction of Semantic Space Models. *Computational Linguistics*, 33(2):161–199.

Ted Pedersen and Anagha Kulkarni. 2006. Automatic cluster stopping with criterion functions and the gap statistic. In *Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: companion volume: demonstrations*, NAACL-Demonstrations '06, pages 276–279, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ted Pedersen. 2006. Unsupervised corpus-based methods for WSD. In *Word Sense Disambiguation: Algorithms and Applications*, pages 133–166. Springer.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Comput. Linguist.*, 24:97–123, March.

Alexander Strehl, Joydeep Ghosh, and Claire Cardie. 2002. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617.

Tim Van de Cruys and Marianna Apidianaki. 2011. Latent semantic word sense induction and disambiguation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1476–1485, Stroudsburg, PA, USA. Association for Computational Linguistics.

Haipeng Zheng, S.R. Kulkarni, and V.H. Poor. 2011. Consensus clustering: The filtered stochastic best-one-element-move algorithm. In *Information Sciences and Systems (CISS), 2011 45th Annual Conference on*, pages 1 –6, march.