Concept-Based Analysis of Scientific Literature

Chen-Tse Tsai, Gourab Kundu, Dan Roth
Department of Computer Science
University of Illinois at Urbana-Champaign
{ctsai12, kundu2, danr}@illinois.edu

ABSTRACT

This paper studies the importance of identifying and categorizing scientific concepts as a way to achieve a deeper understanding of the research literature of a scientific community. To reach this goal, we propose an unsupervised bootstrapping algorithm for identifying and categorizing mentions of concepts. We then propose a new clustering algorithm that uses citations' context as a way to cluster the extracted mentions into coherent concepts. Our evaluation of the algorithms against gold standards shows significant improvement over state-of-the-art results. More importantly, we analyze the computational linguistic literature using the proposed algorithms and show four different ways to summarize and understand the research community which are difficult to obtain using existing techniques.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning

Keywords

Concept recognition; clustering; bootstrapping algorithm

1. INTRODUCTION

The dramatic growth in scientific communities and the publication trace they generate brought with it the challenge of understanding a scientific community – identifying important concepts, trends, key techniques, applications and the relations between them – by analyzing the publication trace. Earlier studies of this question made use of bibliometrics techniques, mostly considering citation graphs [12] and topic models, forming crude topic clustering based on contextual cues [1, 16]. However, these methods cannot address some key questions such as "what methods were developed to solve a particular problem?", "how did these change over the years?", and "what applications have matured enough to be used as components of other applications?"

In this paper, we propose that if we want to achieve a deeper understanding of a scientific community from the paper trace generated by the community, there is a need to better analyze the text itself; there is a need to identify *mentions* of scientific concepts, *categorize* them and cluster them into *coherent concepts*, and study the relations between concepts of various categories. We develop methods to do that, and our evaluation on the ACL text collection reveals interesting observations and insights on the Computational Linguistics scientific community.

The most basic component of our model is an unsupervised algorithm that identifies *mentions* of concepts; in this paper we focus on two categories of concepts: `TECHNIQUE`s and `APPLICATION`s. For example, in the sentence "We apply support vector machines on text classification." our goal is to identify "support vector machines" as a `TECHNIQUE` and "text classification" as an `APPLICATION`. We define the concept extraction problem in a way that is similar to the named entity recognition problem. This instance of the problem was also studied earlier in [5]. Our first contribution is a bootstrapping algorithm [20, 3] that makes use of a small number of per-category pre-specified seeds. The algorithm is used to induce a decision list of features for each category, that is used, in turn, to annotate mentions as belonging to the category and then to extract additional features based on the newly annotated mentions. By iteratively repeating these two steps, we propagate information from a small number of seeds and learn a robust mention identifier. Our results indicate a significant improvement over earlier results in [5].

However, for the purpose of studying a scientific community, we argue that it is essential to attend to the significant variability in the way authors express a given concept. Our notion of a "concept" needs to capture both minor differences in expressing the concept ("SVMs" and "support vector classifiers") and levels of granularity (such as "support vector machines" and "large margin classifiers"). Existing techniques such as topic models [1] do not support the ability to categorize mentions and, as we show, do not allow one to generate tight enough clusters or provide the level of granularity needed to support a careful analysis of the scientific literature. Similarly, naive mention clustering based on lexical similarity also does not support grouping mentions to semantically coherent concepts. Our second contribution is a new clustering algorithm that makes use of *citation contexts* as a way to group concept mentions to meaningful coherent concepts. Given a citation to paper p, we assume that two mentions m_1, m_2 of a given category (e.g., `TECHNIQUE`) appear in the citation context in a way that indicates both m_1 and m_2 are described in p. In this case, we assume a degree of similarity between the two mentions. With this

CIKM '13, October 27 - November 01 2013, San Francisco, CA, USA

http://dx.doi.org/10.1145/2505515.2505613.

as the basis metric for our clustering algorithm, we show that we can group mentions such as "SVM", "support vector machine", and "maximal margin classifiers", which clearly represent the same concept, but are expressed using very different surface forms.

We quantify the performance of our techniques relative to gold standards and then move to study their impact on the analysis of the Computational Linguistics literature. We present four different ways to summarize our understanding of the community. First, we consider scientific trends in the community over a period of thirty years. We compare our context-citation driven concept formation method to topic models and lexical clustering and exhibit it's advantage in identifying trends and accurately pinpointing the emergence of techniques such as topic modeling and of applications such as sentiment analysis. We also quantify the advantage of our clustering method by evaluating it as a method for forecasting the change in scientific trends. The significance of categorizing concepts to `TECHNIQUE` and `APPLICATION` is shown by several studies in which we exhibit the ability to identify what techniques contribute to a given application and how this has changed over time; for example, we accurately identify the emergence of phrase-based methods in machine translation, and that of conditional random fields as a key techniques for tasks such as named entity recognition.

2. RELATED WORK

Several components of our approach have been seen in earlier work. Our mention extraction algorithm addresses the need to categorize mentions in an unsupervised way by using a Bootstrapping algorithm, building on successful earlier applications of this approach to word sense disambiguation [20] and named entity classification [3, 8]. Closer to our application, [19] used bootstrapping to learn generalized names, such as the names of diseases and infectious agents. They argue that identifying generalized names is much harder than identifying conventional names. Our concepts are also generalized names but with greater ambiguity than in the aforementioned case. The work on semantic class induction (e.g., [14, 15, 17]) is also related to ours. [6] proposed a domain-specific semantic class tagger that makes use of a bootstrapping algorithm. In their problem, a given noun phrase can be labeled with different class labels in different contexts, in a setting that is similar to our concepts. Their work is done in the domain of veterinary medicine.

We apply our techniques to study the scientific literature of a research community. Topic models [1, 4] is perhaps the most commonly used technique recently. Many variants of topic models have been proposed in recent years, from the basic model to those that deal with phrases and those that take temporal aspects into account [2, 18].

The key advantage of our approach over topic models is that we deal with different types of concepts and argue that this is necessary to the type of refined analysis we perform. Topic models will not distinguish `TECHNIQUE` from `APPLICATION`. Moreover, topic model do not produce tight enough cluster for our application; as we show, topics typically contain several different concepts and it is thus harder to summarize a specific concept by just considering the clusters produced using topic models. Citation based methods are also used in this domain: [11] make use of "citing sentences" in scientific papers to analyze research trends and understand the impact of research. However, rather than identifying scientific concepts in the whole collection as we do, their analysis is based on a specific paper. Finally, [5] present work that is used as our starting point in this paper: they extract three types of concepts, focus, technique, and application domain by applying a bootstrapping algorithm, as a way to analyze the dynamics of research communities. In our experiments, we use their results as the baseline for our mention extraction approach. However, they use deeper level of sentence analysis – a dependency graph of each sentence, while we only use shallow parsing; nevertheless, our mention identification algorithm performs significantly better. Moreover, they only extract *mentions* of concepts, and does it only from abstracts, while we use full articles and use our identified mentions to further generate concepts which contribute significantly to the refined analysis we present.

3. COMPUTATIONAL APPROACH

This section presents the algorithmic components of our model, the concept mention extraction and the citation-context based concept clustering.

3.1 Concept Mention Extraction

The proposed bootstrapping algorithm is based on the framework presented in [20, 3]. It learns a decision list for each category $y \in \{\texttt{TECHNIQUE}, \texttt{APPLICATION}\}$. A decision list contains representative features for the corresponding category, and is used to predict the category of mentions in unseen data.

Pre-Processing: The training corpus is passed through a chunker [10]. Let X be all noun phrases (NPs) derived from the chunker. We only consider noun phrases as concept mention candidates.

Features: We extract six types of features for each noun phrase. These features are then matched with features in the decision lists to determine if the noun phrase belongs to any category. Note that several of the features we extract here are common in the bootstrapping algorithm for named entity recognition problem [3]. The following features are used, **1.** The words and bigrams of the NP. (e.g., the NP "bootstrapping algorithm" has three features *unigram:bootstrapping*, *unigram:algorithm*, and *bigram:bootstrapping-algorithm*.) **2.** The words and bigrams next to the NP. (e.g., in a sentence "We use bootstrapping algorithm to extract ...", the NP "bootstrapping algorithm" has features *context-uni:l:use*, *context-uni:r:to*, *context-bi:l:we-use*, and *context-bi:r:to-extract*.) **3.** A feature indicates if the spelling of the NP is all capitals. **4.** The closest verb before the NP, and it should be after the previous NP.

Initialization: We seed the decision lists using a small set of feature seeds which are extracted from some concept mentions of each category. The complete set of seeds used in our experiments is shown in Table 1. Then we go through all the documents once to annotate NPs. An NP is annotated as category y if it has at least one feature in the decision list of y (i.e. the seed set of category y). These newly labeled NPs are added into a list L_y, which maintains the labeled noun phrases of category y throughout our algorithm.

Learning: The bootstrapping algorithm iterates through the following three steps:

1. Compute feature scores. The score of a feature f in category y is defined by $score_y(f) = \frac{|x \text{ has } f, \forall x \in L_y|}{|x \text{ has } f, \forall x \in X|}$ Thus, features which are frequently associated with labeled NPs have higher scores than those that are frequent but their

corresponding NPs are not labeled. We calculate scores for all the features associated with labeled NPs in this step. Features which contain any stop word are removed.

2. Select new features. The feature scores are used to select (up to) top k features which are not in the current decision list and have scores above a pre-specified threshold, t_y, for each concept y. That is, to ensure we select the features with high precision predictive quality, we not only just select top k features, but also set a threshold of the score. Moreover, if the number of selected features is less than n, we decrease the threshold by t. This policy ensures it only includes accurate features into the decision lists in the early iterations, and utilizes low-scored features in the later iterations. We find that some features which have low scores are actually useful. Using a small n allows a cautiously selection of features but may increase training time.

3. Annotate noun phrases. We now use the updated decision lists to annotate NPs in the corpus. An NP x which hasn't been annotated in the previous iterations is labeled as category y if the features of x match at least r features in the decision list of category y.

3.2 Citation-Context based Concept Clustering (CitClus)

Recognizing different mentions that refer to the same concept is essential to support meaningful analysis of a community. Some well-known clustering algorithms (e.g., k-means) do not serve our needs here. Because we don't know how many clusters will be there; our goal is to group together mentions of the same concept. Pre-specifying the number of clusters may not generate tight enough clusters.

In this section, we introduce a citation-context based clustering algorithm, which utilizes both citation context information and lexical similarity to group the extracted concept mentions to coherent concepts. The following discussion focuses on a given type of concept (e.g.`TECHNIQUE`). We first define the lexical similarity between two mentions m_1 and m_2:

$$msim(m_1, m_2) = \frac{l_{1,2}}{\max(l_1, l_2)}, \quad (1)$$

where l_i is the number of words in m_i and $l_{i,j}$ is the number of words common to m_i and m_j. The similarity between two clusters of mentions C_1 and C_2 is

$$csim(C_1, C_2) = \frac{\alpha_{1,2}(\delta)}{|C_1|}, \quad (2)$$

where $\alpha_{1,2}(\delta)$ is the number of mentions $m \in C_1$ satisfy $msim(m, m_j) > \delta$, for some $m_j \in C_2$. This similarity represents the proportion of mentions in C_1 which have at least one similar mention in C_2. The clustering algorithm for a given concept is summarized in Algorithm 1. The key step in computing this similarity is that we initialize our clusters to contain mentions that appear in the context of a citation to the same paper. We note that it is possible to consider the context in which a mention occurs, but we have found that considering only citation context is sufficient, and more robust. The intuition is that if multiple mentions are referred to as introduced in paper p, this provides a strong clue of these mention representing the same concept. If $\text{cit}(m)$ is the set of citations that follow the mention of m in an article, then we initialize m_1 and m_2 to be in the same cluster if $cit(m_1) \cap cit(m_2) \geq \theta$. That is, we only use mentions that are followed up by a citation somewhere in our training data, and use the fact that they are followed by the *same* citation as a way to initialize them into the same cluster. Not only this helps grouping mentions into semantically coherent concepts, it also cleans up the prediction results of our bootstrapping mention extractor. In our experiments, we set $\theta = 1$.

Algorithm 1 Concept clustering (CitClus)

1. Group mentions based on the citation context. It generates $\mathcal{C} \leftarrow \{C_1, \cdots, C_n\}$, such that $\forall m_k, m_l \in C_i$, $cit(m_k) \cap cit(m_l) \geq \theta$
2. Clean up clusters. For each mention m in C_i, remove m from C_i if there exists a C_j such that $|C_j| < |C_i|$ and $|msim(m, m_k) > \delta| < |msim(m, m_l) > \delta|$, $\forall m_k \in C_i$, $\forall m_l \in C_j$.
3. Merge the clusters in $\mathcal{C}$ to form the final clustering $\mathcal{C}'$.
$\mathcal{C}' \leftarrow \phi$
while $\mathcal{C}$ is not empty **do**
  $S \leftarrow$ the largest cluster in $\mathcal{C}$
  $S' \leftarrow \{C_i : csim(C_i, S) > \delta, C_i \in \mathcal{C}\}$
  while S' is not empty **do**
    $\mathcal{C} \leftarrow \mathcal{C} - S'$, $S \leftarrow S \cup (\cup_{C \in S'} C)$
    $S' \leftarrow \{C_i : csim(C_i, S) > \delta, C_i \in \mathcal{C}\}$
  end while
  $\mathcal{C}' \leftarrow \mathcal{C}' \cup \{S\}$
end while

The initialization results is a set of clusters, each representing a cited paper with its cited concept mentions. We further clean up the clusters from mentions that are wrongly cited or do not represent important concepts in the cited paper. To do that we remove a mention m from cluster C_i if there exists another cluster C_j, which has fewer mentions than C_i but contains more similar mentions of m than C_i does. This step prevents us from combining different concepts into one cluster in the next step. Finally, we recursively merge the similar clusters based on the similarity between two clusters, which is defined in equation (2).

4. EVALUATION OF ALGORITHMS

We evaluate the performance of our algorithmic components in this section.

4.1 Evaluation of Mention Extraction

We evaluate our approach on the ACL Anthology Network (AAN) Corpus [13], which contains 18,292 full text articles. Because the text documents are converted from pdf files, full text is usually noisy. For the mention extraction evaluation we train on 11,005 abstracts along with the corresponding titles. For evaluation, we use 474 abstracts and titles which were manually annotated by [5], and measure the precision and recall scores of each concept type. We randomly sample 50 abstracts from the test set as the development set. This set is used to select the parameters of the proposed mention extractor and determine the number of iterations. The parameters used in our mention extractor are: $k = 2000, n = 200, r = 2, t_{\texttt{TECHNIQUE}} = t_{\texttt{APPLICATION}} = 0.3$. We initialize the decision lists using 5 feature seeds for `TECHNIQUE` and 9 feature seeds for `APPLICATION`. For `APPLICATION` we seed with names of ACL applications, while for `TECHNIQUE` we use generic terms such as "algorithm" and "model" (see Table 1). At test time we remove stop words from the predicted noun phrases to obtain cleaner results.

We compare our proposed approach with the results of [5], which applies a bootstrapping algorithm to a similar problem, but used a dependency graph representation of sen-

Category	Seeds
TECHNIQUE	unigram:bootstrapping, unigram:model, unigram:approach, unigram:algorithm, unigram:models
APPLICATION	unigram:translation, unigram:transliteration, unigram:mt, unigram:qa, bigram:entity-recognition, unigram:extraction, bigram:question-answering, bigram:sense-disambiguation, unigram:wsd

Table 1: The complete set of seed features which are used in all the experiments for the concept mention extractor.

Approach	TECHNIQUE			APPLICATION		
	Pre.	Rec.	F1	Pre.	Rec.	F1
GM (2011)	30.5	46.7	36.9	27.6	57.5	37.3
Our approach	48.2	48.8	**48.5**	44.0	47.3	**45.6**

Table 2: Comparing the proposed mention extraction algorithm with [5].

Approach	TECHNIQUE	APPLICATION
LexClus	1.72	1.62
CitClus	1.28	1.49

Table 3: A comparison of clustering approaches. Clustering performance is evaluated on manually annotated gold clusters, using *variation of information* (eq. (3)).

tences. That is, instead of using n-gram features as seeds, they use patterns of dependency (sub)graphs as seed patterns. In each bootstrapping iteration, top k patterns in each category are included into their decision lists. The comparison between their method and the proposed approach is shown in Table 2. Note that when comparing the initial decision lists, which only use the seeds, the F1 scores of both approaches are similar. However, our seeds have a much higher precision (and lower recall) than the seed patterns used by [5]. The reason is that we use specific names as seeds, while the dependency tree patterns are more general. However, after running the algorithm, our approach clearly outperforms their method in terms of both precision and recall and leads F1 score by 11% in TECHNIQUE and 8% in APPLICATION. Our analysis indicates that this is due to the careful policies of feature selection and the adaptation step that allows us to continue selecting tail features.

4.2 Evaluation of Concept Clustering

To evaluate the quality of concept clustering, we randomly sampled 1000 full text of articles from the period 2000–2011 in the AAN corpus and manually clustered the mentions recognized by our concept mention extractor, separately for each concept type. To reduce the number of mentions and get a cleaner list of mentions, we only use the mentions which appear, somewhere in the collection, followed by a citation. We further remove clusters with less than four different mentions to focus on important concepts. It results in 17 clusters for APPLICATION and 19 clusters for TECHNIQUE in that time period.

We compare our citation-context based Algorithm 1 (CitClus) with a baseline which only utilizes lexical similarity between mentions. The baseline is derived by running Algorithm 1 without using the information of citation context. That is, it doesn't make use of the first step of Algorithm 1 and it simply views each mention as a single cluster in the beginning. The mentions with high similarity will be merged in the third step of Algorithm 1. We call this baseline LexClus. In both CitClus and LexClus, we set the threshold hold δ of similarities (equation (1) and (2)) to be 0.5.

We use a standard method for clustering evaluation to quantify our results. A clustering is compared with the gold clustering by selecting the cluster which covers most of the mentions in each gold cluster, and then computing the variation of information (VI) [7] between the gold clustering and the selected clusters. The variation of information of two clustering $\mathcal{C}$ and $\mathcal{C}'$ is defined as:

$$VI(\mathcal{C},\mathcal{C}') = H(\mathcal{C}) + H(\mathcal{C}') - 2I(\mathcal{C},\mathcal{C}'), \quad (3)$$

where $H(\mathcal{C})$ is the entropy associated with clustering $\mathcal{C}$ and $I(\mathcal{C},\mathcal{C}')$ is the mutual information between the two clusterings. The lower $VI(\mathcal{C},\mathcal{C}')$ is, the more similar $\mathcal{C}$ and $\mathcal{C}'$ are. Table 3 compares LexClus and CitClus. We can see that CitClus clearly performs better than LexClus, especially on TECHNIQUEs. Carefully analyzing the emerging clusters reveals, as expected, that the lexical similarity metric cannot capture semantic similarity; it doesn't recognize the similarity between "topic modeling" and "latent dirichlet allocation", "Quinlan's c4.5" and "decision tree", and "maximal entropy classifier" and "logistic classifier". In contrast, these pairs of mentions are likely to cite the same paper, thus they will be clustered correctly by a citation-context based clustering algorithm. This also indicates why there was no need to consider other context of mentions; citation context provides a very robust clue.

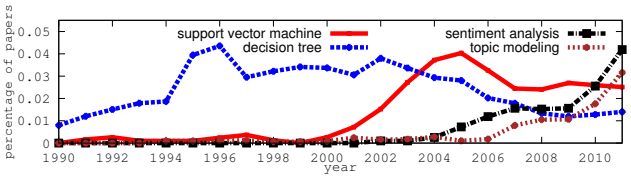
Two experts in the ACL literature generate our gold standard. The inter-annotator agreement is 0.81 for TECHNIQUE and 0.92 for APPLICATION. Most of inter-annotator disagreements are due to concept granularity problems. For example, should "structured prediction" be clustered with SVM, CRF, or perceptron? If we have a single cluster representing structured prediction, we cannot analyze structured prediction accurately without knowing that the other concepts are closely related to it. In the second step of Algorithm 1, we attempt to remove the hierarchical concepts from the clusters in order to focus on lower level concepts. Further study of this would be interesting.

5. UNDERSTANDING COMPUTATIONAL LINGUISTIC RESEARCH

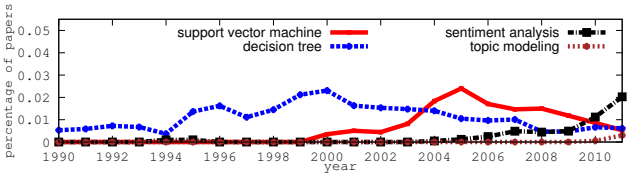
In this section we use our model to study several metrics that indicate an understanding of the research done in the computational linguistics community.

5.1 Trends Analysis

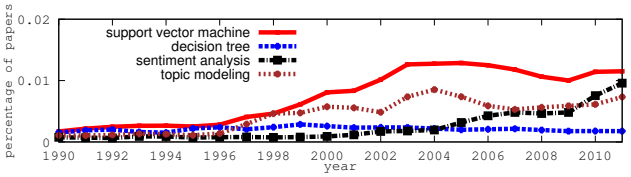
We use the clustered concepts to analyze trends of some important concepts. In this experiment, we use the model trained on abstracts in the previous experiment to extract concept mentions from 15,717 full text articles from the period 1990–2011. We then apply the two concept clustering algorithms, LexClus and CitClus to the extracted concept mentions. We further compare the trends derived by these two concept clustering algorithms with latent dirichlet allocation (LDA) [1] approach. For computing the clusters of LDA we remove all the stop words and numbers and lowercase and stem every word of all the papers. We then use GibbsLDA++ [9] by setting the number of topics to $K = 200$, $\alpha = 50/K$, and $\beta = 0.1$. For a given query concept string, we first select the topic which gives the highest probability to this string. Following the method used in [4], we average θ_{ij} over i in each year, where θ_i is the distribution over topics for document i, and j represents the selected topic. This value is analogous to the fraction of papers which discuss topic j. To compare LDA with the other two con-



(a) The trends obtained from CitClus. The y-axis represents the percentage of papers.



(b) The trends derived from LexClus which only uses lexical similarity to cluster mentions.



(c) The results of LDA. The y-axis is the averaged probability of the selected topic over documents.

Figure 1: A comparison of trends derived by three clustering approaches.

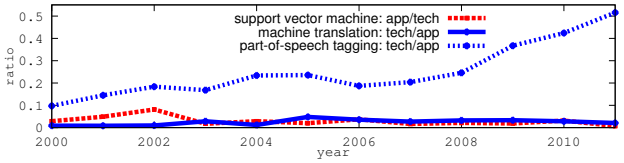


Figure 2: The ratio between number of technique and application mentions for three concepts.

Approach	SVM	DT	TM	SA
LexClus	0.97	0.83	0.73	0.48
CitClus	0.52	0.37	0.37	0.46

Table 4: The predictive quality for four concepts showed in Figure 1. CitClus has lower relative errors in all cases.

cept clustering approaches, we first find the concept cluster which contains the largest number of similar mentions of the query string, and call it the query cluster. We define a paper to be one that focuses on the query concept if most of the extracted mentions in it are also in the query cluster. Therefore, we can derive the fraction of papers which discuss the query concept for each year. Figure 1 shows the trends of four representative concepts. Note that we do not distinguish the categories (`TECHNIQUE` or `APPLICATION`) here, since LDA does not label the topics.

Figure 1a clearly shows that the curves obtained from CitClus nicely capture several important trends. For example, the rise of support vector machine from late 1990s, a huge boost of topic modeling research in recent years, and the recent emergence of sentiment analysis. Comparing Figure 1a with Figure 1b, we see that the numbers of papers obtained by LexClus is much smaller than the numbers identified by CitClus. This results in some misleading trends, as shown for support vector machine and topic modeling. In particular, the LexClus graph shows a sharp decrease in support vector machines starting in 2008 and a slight increase in topic modeling, only starting in 2010. Note that LexClus is a general case of not doing concept clustering at all. If we set a very high threshold to the lexical similarity, the result will be the same as counting the number of the query mention instead of counting the number of mentions in the query cluster. In this case, we will only get fewer papers in each year thus the resulting trends will still be misleading.

Although the LDA graph captures the growth of support vector machine and of sentiment analysis, most of the curves in Figure 1c are very different from those in Figure 1a. Since the selected topics may not represent the given concept only, but rather include other concepts as well, we can see, for example that the curve of topic modeling is already very high before this research area emerged. Carefully looking at the clusters we observed that the cluster corresponding to topic modeling also contains "generative model", which is one reason for these inaccuracies.

5.2 Predictive Quality

We further quantitatively evaluate LexClus and CitClus by considering the predictive power of the resulting trends. For the four concepts in Figure 1, we ask whether we can predict the number of papers in year i, given the number of papers in the previous three years. We apply linear regression to every three consecutive years and use it to predict the value in the fourth year. We omit from this statistics time periods that precede the development of the concept; for instance, topic modeling and sentiment analysis are almost zero before 2000 and this isn't taken into account. Then the averaged relative errors for each concept is calculated by $\frac{1}{2012-j}\sum_{i=j}^{2011}\frac{|\hat{y}_i-y_i|}{y_i}$, where $\hat{y}_i$ is the prediction in year i, y_i is the true value obtained from clustering, and j is the year which $y_i = 0, \forall i < j$. Note that we do not have a notion of ground truth, so each prediction $\hat{y}_i$ only compares with the true value y_i derived by the corresponding clustering. The intuition is that the better the grouping of mentions into coherent concepts is, the more stable the trend graph is, the more accurate the prediction is. Table 4 shows the results. As expected, CitClus has a much lower relative error in most cases, indicating that trends obtained by CitClus are more predictive and stable.

5.3 Relations Between Concept Categories

Classifying concepts into two categories is useful in several ways. Because a concept may belong to different categories in different contexts, we can study a given concept by analyzing, for example, the ratio between the number of times it appears as an application and the number of times it appears as a technique. In Figure 2, we show three important concepts studied in the ACL community which exhibit very different behaviors. For the curve of support vector machines, we plot it by the number of applications divided by the number of techniques. This curve stays at the bottom, indicating that support vector machines always serve as a technique in the ACL community. In contrast, since machine translation is always a popular application, the ratio: number of techniques over applications always stays low. Finally, the curve of part-of-speech tagging grows through the years. It indicates the maturity of this concept, as more and more people use part-of-speech tagging as a technique in recent years. However, there are still many papers aim at solving semi-supervised or unsupervised part-of-speech tagging problems, it still has a significant presence also as an application.

In addition to studying different categories within a given

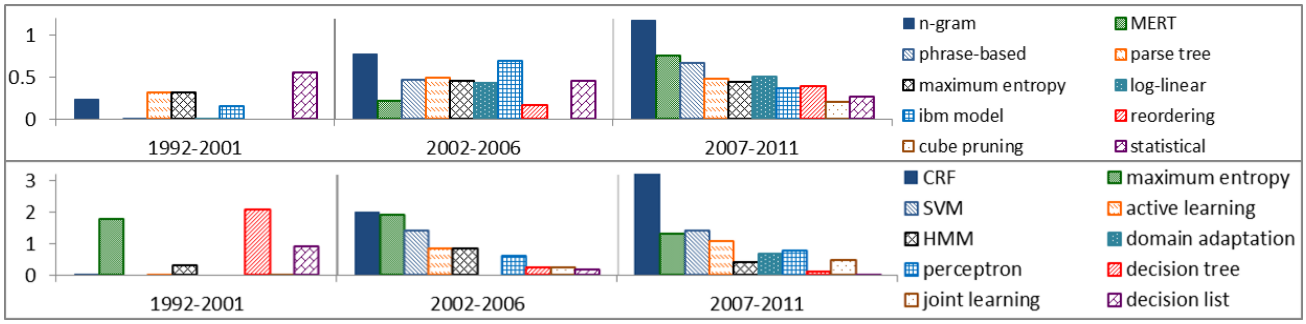


Figure 3: Ten selected techniques used in machine translation (top) and in named entity recognition (bottom). The y-axis is the number of concept mentions normalized by the number of selected papers.

concept, another natural question is to consider how different concepts that belong to different categories interact with each other. Here we study the "apply" relationship between different types of concepts. More specifically, for a given application, we are interested in the techniques which have been applied to it. We first use the same method as in the previous experiments to select the papers that focus on the given application. Then, among all the technique concepts extracted from those papers, we picked the ten most popular concepts and analyzed the change in their usage over time. Figure 3a shows ten techniques used in machine translation and clearly captures the growth of phrase-based and minimum error rate training in the last five years. In Figure 3b, decision tree and decision lists are shown to be popular for the named entity recognition problem early on, only to disappear after 2007. We can also see the obvious rise of conditional random fields after 2002 and it dominates other techniques after 2007.

6. CONCLUSION

This work proposed algorithmic tools for identifying, categorizing and clustering mentions of scientific concepts. We showed that these tools can provide a rather deep understanding and useful insights into the progress, changes and trends of research in the ACL community. This work opens up a range of questions from algorithmic issues for improving our approaches to better deal with hierarchies, to understand concepts that are being used across communities.

Acknowledgments: This work is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20155. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

7. REFERENCES

[1] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *JMLR*, 2003.

[2] D. M. Blei and J. D. Lafferty. Dynamic topic models. In *ICML*, 2006.

[3] M. Collins and Y. Singer. Unsupervised models for named entity classification. In *EMNLP*, 1999.

[4] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proc Natl Acad Sci U S A*, 2004.

[5] S. Gupta and C. D. Manning. Analyzing the dynamics of research by extracting key aspects of scientific papers. In *IJCNLP*, 2011.

[6] R. Huang and E. Riloff. Inducing domain-specific semantic class taggers from (almost) nothing. In *ACL*, 2010.

[7] M. Meilă. Comparing clusterings by the variation of information. *Learning theory and kernel machines*, 2003.

[8] C. Niu, W. Li, J. Ding, and R. K. Srihari. A bootstrapping approach to named entity classification using successive learners. In *ACL*, 2003.

[9] X.-H. Phan and C.-T. Nguyen. Gibbslda++: A C/C++ implementation of latent dirichlet allocation (LDA). 2007.

[10] V. Punyakanok and D. Roth. The use of classifiers in sequential inference. In *NIPS*, 2001.

[11] D. Radev and A. Abu-Jbara. Rediscovering acl discoveries through the lens of acl anthology network citing sentences. In *ACL*, 2012.

[12] D. R. Radev, M. T. Joseph, B. Gibson, and P. Muthukrishnan. A bibliometric and network analysis of the field of computational linguistics. *JASIST*, 2009.

[13] D. R. Radev, P. Muthukrishnan, and V. Qazvinian. The ACL anthology network corpus. In *Proceedings of the 2009 Workshop on Text and Citation Analysis for Scholarly Digital Libraries*, 2009.

[14] E. Riloff and J. Shepherd. A corpus-based approach for building semantic lexicons. In *EMNLP*, 1997.

[15] B. Roark and E. Charniak. Noun-phrase co-occurrence statistics for semiautomatic semantic lexicon construction. In *ACL*, 1998.

[16] Y. Sim, N. A. Smith, and D. A. Smith. Discovering factions in the computational linguistics community. In *ACL*, 2012.

[17] M. Thelen and E. Riloff. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *EMNLP*, 2002.

[18] X. Wang, A. McCallum, and X. Wei. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *ICDM*, 2007.

[19] R. Yangarber, W. Lin, and R. Grishman. Unsupervised learning of generalized names. In *ACL*, 2002.

[20] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervied methods. In *ACL*, 1995.