Analysis of Top-Tier Computer Science Conferences and Research Communities using NLP

Ethan Li and James Hong

December 16, 2016

Abstract

Computer science is a broad field with many sub-topics and disciplines, each with their own top-tier conference venues and distinct research communities. This project aims to study the structure and characteristics of these research communities through paper titles and abstracts published in top conferences using techniques from natural language processing and understanding, AI, and ML. By attempting the task of predicting the conferences or research communities associated with published papers, we identify interesting insights about research communities reflected in the semantic and syntactical content of paper titles and abstracts.

1 Introduction

Computer science is a very broad subject with many smaller fields and disciplines. Most researchers are very specialized in areas such as deep learning, databases, operating systems, and HCI. These sub-disciplines encompass a range of topics in themselves. Many researchers and students never reach out beyond their own specialized fields, and, in doing so, forgo interesting knowledge and possibilities for research that spans several fields. Likewise, in pigeonholing oneself into a single field or even a larger area such as systems, AI, or theory, one potentially hampers one's own ability to appreciate and evaluate the work of others. This can lead to very innovative papers being rejected (because reviewers lack expertise to review them) or mediocre work being accepted at top-tier conferences where the main focus differs from the content of the paper. Tackling the lack of pervasive and rigorous mixing between CS disciplines is a challenging problem.

We propose a simpler task to apply NLP tools and algorithms to understand the relationship between varying disciplines in CS. In doing so, we hope to detect patterns in CS publications and examine whether one can objectively capture the attributes that make a paper suitable for one CS conference and not another.

2 Background & Related Work

Prior investigations into the structure of research communities, including but not limited to the publication venue prediction task, have proposed and applied methods for mining citation or authorship network connectivities and for modeling topics from document texts. Our review of related work focuses on the latter approach, as the analysis of document texts which this approach accomplishes complements the former approach.

Early approaches for topic modeling in our problem domain identified latent topics between words, authors and documents [11]. One line of investigation applied Latent Dirichlet Allocation (LDA) to associate latent topics between article titles and conferences in order to characterize topics of different conferences and to identify relationships between conferences [2, 3]. This approach was extended to associate authors with topics in order to detect semantically-related research communities [1, 7]. Similar approaches added a latent subject layer for conferences and associated subjects with authors and topics [12], incorporated cited authors into models of topics and conferences [14], or used part-of-speech patterns to identify n-gram multiwords for topic extraction [5]. Unlike these previous investigations, we do not apply topic modeling of titles for the venue prediction task, but rather we attempt to exploit latent semantic and syntactical structure in titles and abstracts.

The most similar prior investigation to our project focuses on engineering lexical and syntactical features to capture writing style for venue prediction from abstracts [13]. However, rather than hand-crafting feature sets to represent document-level lexical and syntactical properties, we use word embeddings to generate feature vectors for downstream use. As these word embeddings capture semantic and syntactical relationships between different words, our approach aims to combine the advantages of the previously discussed work in topic modeling and writing style modeling for venue prediction.

Unsupervised word vector embedding techniques such as word2vec [8] and GloVe [9] map words in a corpus onto a vector space such that words with common contexts in the corpus have similar vectors. This property makes such embeddings useful for capturing semantic and syntactical relationships between words.

3 Dataset

For our tasks, we are using the ACM-Citation-Network v8 dataset (henceforth referred to as *ACM-Full*), which consists of CS paper citations compiled by Tang, et al. [11]. In total, the dataset contains citations for publications in 273,275 venues, many of which are actually book titles or conferences from different years. To make the problem human-understandable, we only included citations with both a title and an abstract, and we filtered the publication venues to 31 top-ranked conferences, with each conference aggregated by year. We refer to the resulting dataset as *ACM-Conf*. To reduce training set imbalance, we limit each conference to 1000 examples, though many conferences still have far fewer training examples. This yields 35,168 training examples and 1,483 each for validation and test. Refer to Appendix 11.1 for the labels and their train and test set sizes.

Because many conferences are very similar (e.g., ACL and NAACL), we built another dataset, *ACM-Field*, by aggregating conferences from *ACM-Conf* into disciplines for our labels; this produces more balanced training labels and is more focused on the task of classifying topics in CS. There are 14 disciplines, including AI, computer vision, NLP, HCI, OS, mobile, theory, etc. In total, we have 26,009 examples for training and 700 each for validation and testing. Refer to Appendix 11.1 for the full set of labels and their train and test set sizes. Examples of article titles in this dataset include *Efficient online structured output learning for keypoint-based object tracking* in computer vision, *Using model checking to debug device firmware* in operating systems, and *New hardness results for congestion minimization and machine scheduling* in theory.

4 Methodology

Our main project focuses on classification of paper titles and abstracts into either conferences or CS subfields.

4.1 Oracle: Human Baseline

Our oracle consists of manual prediction by team members of unprocessed documents. For *Task 1*, we attained 61% accuracy on 100 randomly-sampled articles. For *Task 2* with *ACM-Field*, we attained 69% accuracy on 650 randomly-sampled articles; we noticed that confusion between the AI, ML, CV, and data mining labels was a major source of human classification error.

4.2 Models

We experimented with several classifiers including softmax, kernelized SVM, k-NN, and multi-layer perception (MLP). These models take an input vector consisting of features and generally lose sequence information. To remedy this, we tested a recurrent model with gated recurrent units (GRU).

Of these models, the softmax classifier is simplest to tune and performs very well. We attribute its high accuracy to the expressiveness of the features. Moreover, the softmax classifier is quick to train using SGD. By contrast, the kernelized SVM with a Gaussian (RBF) kernel trained very slowly and returned similar, but slightly, worse accuracies. Likewise, the MLP classifier took much longer to train and did not outperform softmax. While, it is likely that investing significant effort in tuning the hyperparameters of these non-linear models would yield better performance, the softmax baseline is very strong by itself. We report the full accuracies in the following sections.

Whereas a feed forward neural network such as the MLP is deep with a fixed depth, a recurrent neural network (RNN) is deep with respect to timesteps, namely tokens in the sequence. This allows RNN architectures to take variable length inputs and retain sequence information. We experimented with the GRU variant of RNNs, and our experiments motivate similar findings as the above with softmax: the expressiveness of the word embeddings plays a larger role in the classification results than the choice of model.

4.3 Features

Baselines: Unigrams & Bigrams

For our baseline, we used unigram and bigram features. To avoid exploding the feature space, we threshold the set of features to include only those with support of at least 50. This yields 8000 unigrams and 25,000 bigrams. While even a simple model performs well with unigram features, we show that the large number of features easily results in overfitting to the training data.

Word Vectors

A second approach that we tried was word vectors, trained from scratch using the word2vec algorithm on the full ACM citations dataset as well as pre-trained vectors using GloVe. The reason we used word2vec rather than GloVe for training custom vectors is that the former can be extended to obtain paragraph vectors, which represent variable length text in a single vector. These are described below in the next section.

There are many ways to aggregate word vectors to obtain an embedding for a text. We experimented with both maximizing and averaging over word vectors of the tokens in a text. These yielded nearly identical results. We also evaluated a variety of model parameters to train the word embeddings, but these also yielded nearly identical results.

Paragraph Vectors

Paragraph vectors were introduced in 2014 by Quoc Le and Tomas Mikolov [6], when they demonstrated state-of-the-art classification accuracy on two hotly contested sentiment analysis tasks. The algorithm works by introducing a special token into each context of a text to represent the text as a whole. In doing so, the model can capture information about all of the contexts and sequence information. Figure 1 shows an example of how the distributed bag of words paragraph vector algorithm works. Like word2vec, paragraph vectors can also be trained using the skipgram architecture. One challenge with paragraph vectors is the difficulty of tuning them; it is difficult to replicate their performance and, as far as we know, no one has successfully replicated the reported accuracies on the sentiment treebank dataset [10, 4].

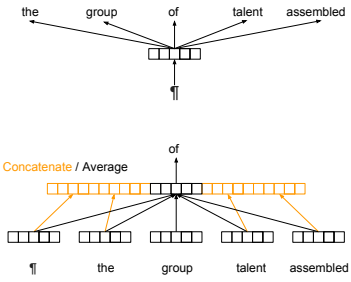


Figure 1: The paragraph vector algorithm introduces a special token into each context, but is analogous to CBoW and skip-gram otherwise. Image source [4].

Parts-of-Speech

One interesting question that we might ask about varying publication venues is whether there are differences in writing style. One simple approximation of this is to use only part-of-speech information. To extract

these features, we used a basic perceptron POS tagger in NLTK to tag our input sequences. From this, we extracted POS unigrams and bigrams. While accuracy using only POS features is mediocre compared to that of word vectors or n-grams, we did find some interesting patterns.

Combined Features

It's often the case that concatenating vectors of independent features boosts classification accuracy. We tested this cautiously, as unigrams and bigrams are very high dimensional to begin with. Our model easily overfits with either alone. A more interesting experiment is concatenating differently trained word vectors (CBOW & skipgram) and features such as POS. Overall, we find that concatenating does not improve performance noticeably. Word vectors already embed much of the information about the tokens in the text.

5 Task 1: Conference Classification

5.1 Definition

Given a query article's title and abstract as unstructured text, our system will label it with a prediction of which conference, out of the 31 conferences in *ACM-Conf*, the article is published in. For instance, *Increasing Datacenter Network Utilisation with GRIN* would have label NSDI, while *Optimistic Crash Consistency* would have label SOSP (note, we have omitted the abstracts here for brevity). One disadvantage of this task is that incorrect predictions for similar conferences are penalized equally. For example *Millions of Little Minions: Using Packets for Low Latency Network Programming and Visibility* was published at SIGCOMM, but could be easily mispredicted as NSDI. This is especially problematic for conferences in OS (SOSP and OSDI) which alternate between years, and conferences such as NAACL and ACL which differ primarily by region, with the former being "North American".

5.2 Results & Analysis

Features	Train	Dev	Test
Human (Oracle)			**61.0%**
Random			3.2%
Unigram (Baseline)	91.6%	51.7%	52.7%
Bigram (Baseline)	99.9%	46.7%	46.8%
GloVe (Wiki+Gigaword)	56.5%	43.1%	44.2%
Word2Vec (ACM)	65.1%	51.9%	49.8%
Paragraph Vectors	38.0%	24.0%	25.2%

Table 1: Task 1 accuracy by feature type with a softmax classifier. Low-dimensional features, such as word vectors, are less susceptible to overfitting.

On the conference prediction task, we obtain 52.7% accuracy using just unigram features. The model very clearly overfits with 91.6% accuracy on training data. Moreover, this result is incorrigible even with very aggressive regularization. Also, this result is shy of our human baseline. One possible reason for why unigrams outperform the other evaluated features is that when predicting among conferences, many of which are similar, the model will learn that certain conferences emphasize certain topics. For instance, whereas NSDI and IMC (which is subsumed by the SIGCOMM label) are both networking conferences, the former deals with networked systems implementation while the latter focuses on Internet measurement. This means that terminology related to Internet governance such as "autonomous system (AS)" would appear in the latter but not the former. Unigrams and bigrams can explicitly capture these features, whereas techniques such as word2vec and GloVe are less direct. Our training accuracies with GloVe and word2vec suggest that this indirection has a regularizing effect in addition to having an negative impact on overall accuracy.

Figure 2 shows the confusion matrices for test for the softmax classifier trained with word2vec embeddings. First, many of the conferences in the data set are dataset are too small and only rarely

predicted. For instance, ICFP, International Conference on Functional Programming, has very few training examples. Oversampling might help here but it is also possible that works published in ICFP can also be published in larger programming language conferences such as POPL and PLDI. Second, the results indicate that the model does in fact confuse conferences that are similar (see ICCV and CVPR; ACL, NAACL, and EMNLP; SIGCOMM and NSDI). Indeed, may researchers who submit to these conference groups choose to do so based on whichever deadline is nearest and annually attend each one.

When we modify the scoring mechanism so that if the true label is captured in the top 2 and 3 classes predicted, the test accuracy with word2vec embeddings rises to 72.7% and 83.0%, respectively. This result corroborates our observation that confusion between related conferences in the same CS subfield is a major source of classification error and motivates our second task, prediction of CS research subfield rather than conference venue.

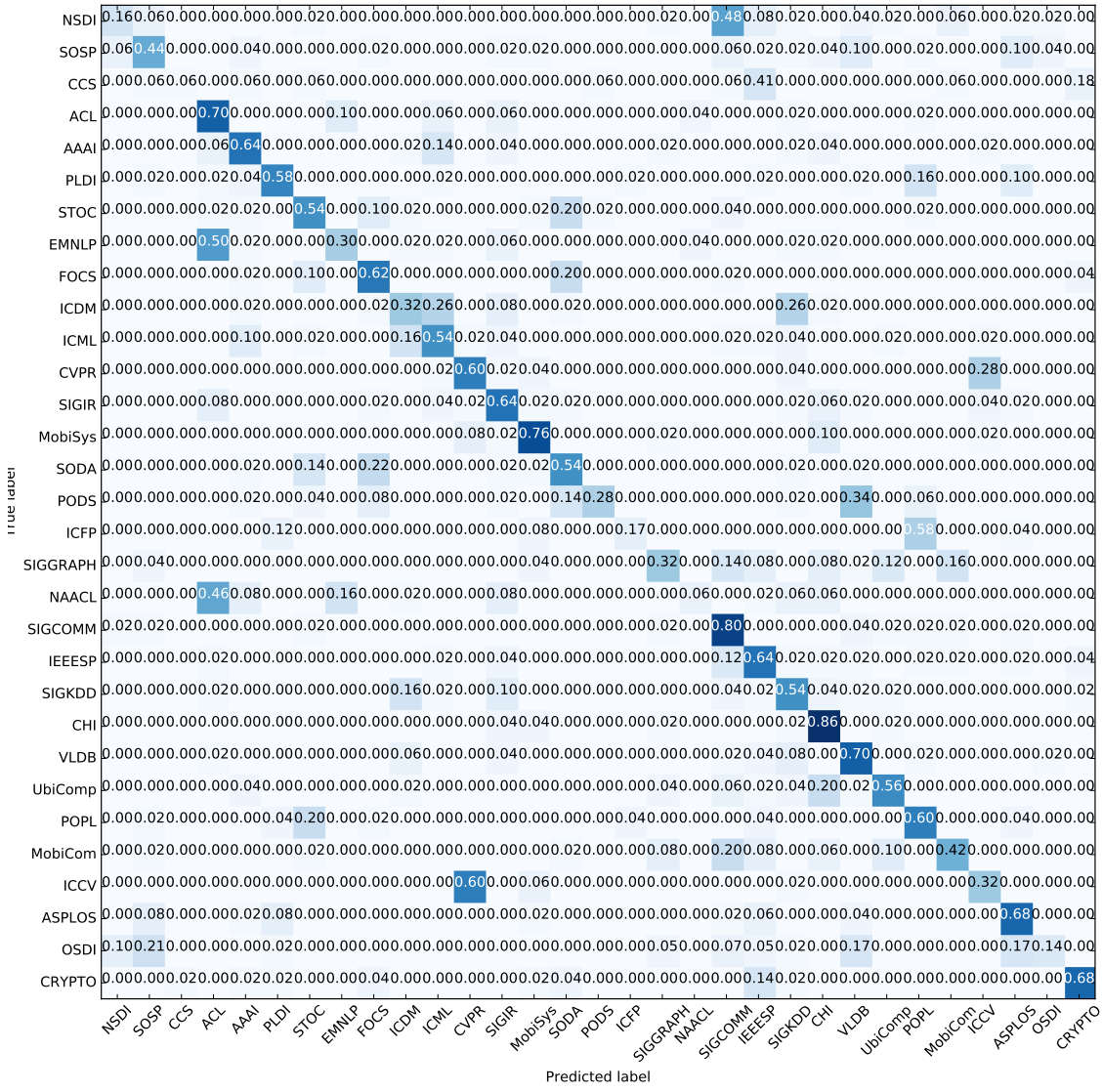


Figure 2: Test confusion matrix with Word2Vec features on *ACM-Conf*. Only test is shown for space reasons.

6 Task 2: Field Classification

6.1 Definition

Given a query article's title and abstract as unstructured text, our system will label it with a prediction of which CS research subfield (e.g. graphics, systems, HCI), out of the 14 subfields in *ACM-Field*, the article is published in. This addresses the similar conference problem from task 1 described in section 5. For example, similar conferences such as SIGCOMM and NSDI fall under the networking category. Likewise, ACL, NAACL, and EMNLP are grouped under NLP.

6.2 Results & Analysis

From our results for predicting CS subfield, we find that unigrams are a very strong baseline, with a test accuracy of 73.3% (Table 2), with overfitting a less severe issue than in Task 1. The strong performance by unigram features may be explained by the same reasons as we proposed in Task 1. While they capture many features and overfit to training data, they perform decently on dev and test and outperform the oracle, which had a test accuracy of 69%.

Using word vectors appears to have a regularizing effect as in Task 1, though the loss in test accuracy with pretrained GloVe vectors, which achieve 65% test accuracy, is more than made up for with word2vec vectors trained on our dataset, which achieve 77.6% test accuracy and outperform the other features evaluated on this task. Thus, the pretrained vectors do not appear to generalize well, perhaps due to the importance of specialized CS research terminology in our texts, while word vectors trained on CS research texts do outperform the baseline.

Features capturing word sequences do not appear to outperform the mean word vector feature with word2vec vectors trained on our corpus. Paragraph vectors significantly underperform in comparison to other methods. Given past difficulty in replicating the high performance previously reported for paragraph vectors on other tasks, this result may indicate that our paragraph vector embeddings are not fully tuned. The GRU model, which uses the full word vector sequence for each document rather than the mean word vector of a document, approaches but does not surpass the test accuracy of the mean word vectors with a softmax classifier; GRU models of varying internal projection dimensions from 16 to 128 all appeared to converge to the same test accuracy. While further tuning of the GRU architecture may produce improvements in test accuracy, it appears that capturing sequential structure of words does not provide immediate significant improvements over the mean word vector representation.

As shown in Table 3, the softmax model with word2vec embeddings performs at least as well as more complex models with word2vec embeddings while also maintaining less overfitting. Thus, it appears that use of word vector embeddings to represent words in a document, rather than the choice of model for combining word vectors into a vector to represent the document, is the most powerful contributor to test accuracy identified in this project.

The confusion matrices for the mean word2vec features with the softmax model (Fig. 3) show misclassification errors between AI, ML, and information retrieval, and between mobile and OS. These errors are similar to the errors made by our oracle, suggesting that the these subfields overlap significantly in the types of articles that are published to them. Consistent with this conclusion, modifying the scoring mechanism so that the true label is captured in the top 2 and 3 classes predicted increases test accuracy to 89.4% and 94.4%, respectively (Table 4).

Features	Train	Dev	Test
Human (Oracle)			69.0%
Random			7.1%
Unigram (Baseline)	83.1%	73.7%	73.3%
Bigram (Baseline)	97.7%	69.0%	68.2%
GloVe (Wiki+Gigaword)	71.0%	65.0%	65.0%
Word2Vec (ACM)	78.3%	75.4%	**77.6%**
Paragraph Vectors	42.9%	39.1%	41.4%
Word2Vec-Seq (GRU)	82.1%	70.1%	75.1%

Table 2: Task 2 accuracy by feature type. Except for the oracle and Word2Vec-Seq (GRU), all feature types use linear softmax classifier; Word2Vec-Seq (GRU) uses a GRU classifier.

Model	Train	Dev	Test
Softmax	78.3%	75.4%	**77.6%**
SVM (RBF kernel)	81.5%	78.1%	76.1%
MLP	100%	78.5%	76.9%
GRU	82.1%	70.1%	75.1%

Table 3: Task 2 accuracy by model type using word vectors trained on our corpus. Except for the GRUs, all model types use the mean of word vectors for all words in a document; GRU uses the full sequence of word vectors in the document.

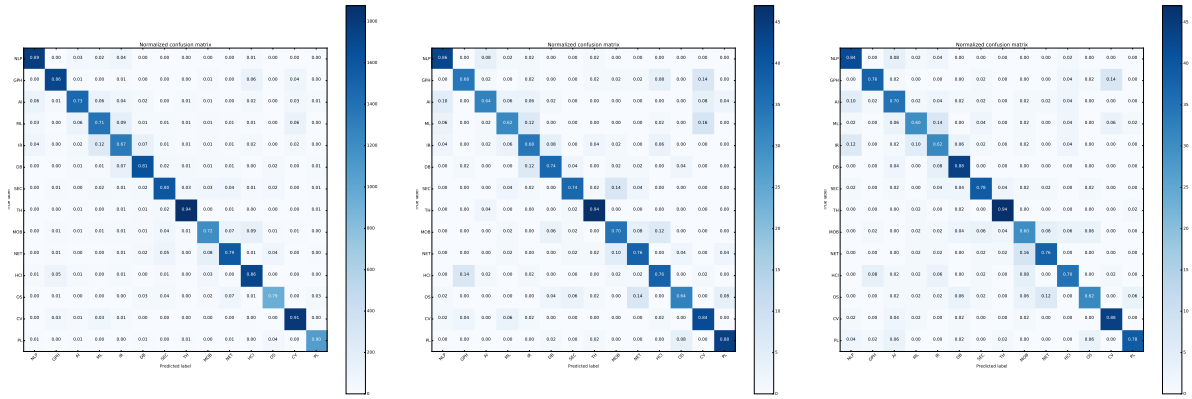


Figure 3: Train, dev, test confusion matrix with mean word2vec features and the softmax model on *ACM-Field*.

Top k	Train	Dev	Test
1	78.3%	75.4%	77.6%
2	93.7%	90.9%	89.4%
3	97.2%	95.6%	94.4%
4	98.6%	96.6%	97.3%
5	99.2%	97.6%	98.1%

Table 4: Train, dev, and test accuracies for the mean word vector embedding with the softmax model, when a correct prediction appears in the top k predictions for each document.

7 Task 3: Exploration

7.1 Part-of-Speech Features

As expected, POS features performed worse than word features (Table 5). However, the results are still interesting: we found that POS features were able to identify theory and HCI papers with greater-than-expected precision and recall (Fig. 4). We believe that this is because the writing style in those fields is sufficiently different from the rest. Note that POS unigrams perform poorly compared to POS bigrams, so we only report results for the latter.

Features	Train	Dev	Test
POS Unigram	25.8%	24.7%	24.8%
POS Bigram	39.7%	32.1%	30.8%

Table 5: Accuracy with only POS information and a softmax classifier on Task 1.

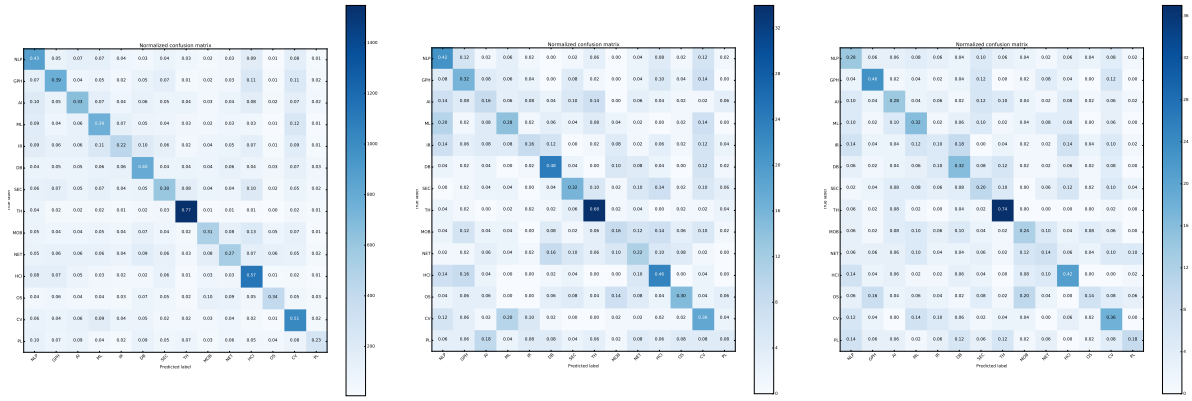


Figure 4: Train, dev, test confusion matrix on *ACM-Field*. POS bigram features strongly identify theory papers.

7.2 Pairwise Classification

One of the original questions we set out to answer is whether different conferences can be classified based on natural language features. This task should be trivial for conferences in two separate fields. For instance, a logistic regression classifier easily obtains 98% accuracy separating computer vision and NLP. Likewise, accuracy is still high when the comparison has slightly more overlap such as with HCI and graphics, where a simple classifier obtains 91.0% accuracy. The slight loss of accuracy can be attributed to the fact that topics such as visualization appear in both communities.

Some more interesting comparisons to draw are between very similar conferences. Table 6 shows the accuracies from the aforementioned comparisons as well as ones between similar conferences. For example, ACL and EMNLP are not well separated, though the classifier still exceeds random chance. The same is true for SOSP and OSDI, the top two OS conferences, though it is possible that this result is confounded by the lack of data in general as both SOSP and OSDI are much smaller and have far fewer papers than say CVPR. However, AAAI and ICML are more different and therefore separated better, with 80% test accuracy. Finally, the most surprising result we found was that the simple model performs well when classifying between FOCS and SODA, Foundations of CS and Symposium on Discrete Algorithms, respectively. One implication of this finding is that our labels for theory may be too coarse.

Classes	Train	Dev	Test
CV vs. NLP	99.3%	98.0%	98.0%
CHI vs. SIGGRAPH	92.6%	88.0%	91.0%
ACL vs. EMNLP	71.8%	67.0%	65.0%
ICML vs. AAAI	90.0%	86.0%	80.0%
SOSP vs. OSDI	100%	67.4%	68.5%
FOCS vs. SODA	96.1%	82.0%	78.0%

Table 6: Accuracies of binary classification using logistic regression with unigram features

7.3 Unsupervised Clustering

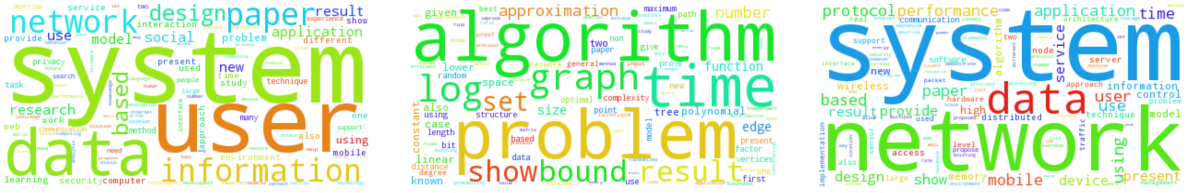
We also experimented with unsupervised learning to investigate whether the conference and field labels are natural clusters within the data, at least through the lens of natural language features. Specifically, we ran k-means clustering with $k = 8$, purposely set to be less than the number of labels in order to force mixing of classes. We used relatively low dimensional (50d) GloVe vectors on this task, as higher dimensionality has an effect on the distance metric. Our results in this respect are interesting and we display three particular clusters in Table 7. In order to provide more intuition as to the meaning of these clusters, we also show their corresponding word clouds in Figure 5.

Cluster	#1	#2	#3	#4	#5
1	HCI (27%)	SEC (15%)	MOB (11%)	IR (10%)	AI (12%)
3	TH (79%)	SEC (5%)	DB (3%)	PL (2%)	ML (1%)
5	NET (25%)	MOB (19%)	OS (18%)	SEC (11%)	DB (11%)

Table 7: Top 5 classes and proportions of 3 clusters from k-Means with $k = 8$.

With the exception of theory clusters, our results show that the natural clustering based on words in the text is less pure than we might initially hope for. This is expected however as the feature extractor we used simply averaged the word vectors without any regard to order or to identifying sequences. We might expect a different result if we had trained an LSTM or GRU autoencoder, which ingests the full sequence and makes predictions until the end of the sequence, and had used the last hidden state. However, it is interesting to see clusters around systems and users and also systems and networks.

Again, theory was the exception (favoring very pure clusters) and we can perhaps see why; words such as algorithm, time and log(arithm) feature prominently. Interestingly, the clusters also capture different uses of log in security and database systems.



(a) Cluster 1. (b) Cluster 3. (c) Cluster 5.

Figure 5: Selected word clouds for the 8-clusters outputted by k-means.

8 Discussion

Our results are interesting for many reasons. First, we found that it is possible to obtain reasonably good accuracy when predicting publication venues from a title and abstract. This can be useful when one would like to decide which conference to submit their latest research to. Second, our results suggest that it is not

possible to do much better than 77% on the 14-way subfield prediction task because the subfields themselves are overlapping; indeed, when we predict more than 1 field, the accuracy quickly jumps to the 90% range.

An orthogonal, but also fascinating, problem that our work does not solve is that of how to divide a complex field such as computer science into distinct fields. Our clustering experiments and the model's confusion in tasks 1 & 2 between AI, ML, and information retrieval are a testament to this. Perhaps training on full papers would solve this problem, but then the likely features will be section headers and overall length, which are less interesting from a natural language understanding perspective. Another challenge is the size of these communities. Whereas conferences such as EMNLP accept over 250 papers annually, SOSP accepts fewer than 30 papers (and only every other year). These disparities make learning inherently challenging. Finally, an unanswered question pertinent to our project is whether communities should be divided based on human understanding of the topics (CV, NLP, HCI, etc.) or extraction of latent topics (as in prior work with topic modeling), or when some research communities grow too large relative to others.

When experimenting, we also tested an alternative division of classes loosely based off Stanford's CS tracks; the 7 resulting classes are listed in appendix under *ACM-Track*. As expected, reducing the number of classes increased accuracy. However, like our 14 class problem, these classes are in no way disjoint when it comes publications, nor are they natural with respect to clustering using the word vector approach in section 7.

9 Conclusions

Ultimately, our results demonstrate that there is sufficient structure and commonality between papers in a field that allows natural language features to be used with reasonable results on classification tasks. These results exceed our own ability (human oracle) to classify, suggesting that we too have much to learn with respect to the varying topics in CS. This is a good thing.

10 Future Work

Our project can be extended in many ways. First, we can apply recursive neural-networks to the part of speech features alone. We see that the performance is low for train, validation and test. This can be attributed to the features not being expressive enough or due to the model. Another possible improvement is to use a more powerful part of speech tagger than the perception one. Second, we can study the structure of the larger communities which have far over 2,000 articles: for instance, the database, graphics, and combined AI/ML/NLP/CV communities. Clustering within these might yield further interesting structures. Thirdly, we have not used publication date as a feature in any of our experiments. It would be interesting to see for the large groups, in particular, how cutting-edge research topics have shifted over time.

References

[1] A. Daud. A topic modeling approach for research community mining. In *5th International Conference on Computer Sciences and Convergence Information Technology*, pages 1078–1083, Nov 2010.

[2] Ali Daud, Juanzi Li, Lizhu Zhou, and Faqir Muhammad. *Conference Mining via Generalized Topic Modeling*, pages 244–259. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[3] Ali Daud and Faqir Muhammad. Group topic modeling for academic knowledge discovery. *Applied Intelligence*, 36(4):870–886, 2012.

[4] Michael Fang James Hong. Sentiment analysis with deeply learned distributed representations of variable length texts, 2015.

[5] Nikhil Johri, Dan Roth, and Yuancheng Tu. Experts' retrieval with multiword-enhanced author topic model. In *Proceedings of the NAACL HLT 2010 Workshop on Semantic Search*, pages 10–18, Los Angeles, California, June 2010. Association for Computational Linguistics.

[6] Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents.

[7] Daifeng Li, Ying Ding, Xin Shuai, Johan Bollen, Jie Tang, Shanshan Chen, Jiayi Zhu, and Guilherme Rocha. Adding community and dynamic to topic models. *Journal of Informetrics*, 6(2):237 – 253, 2012.

[8] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[9] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation.

[10] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. Citeseer.

[11] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: Extraction and mining of academic social networks. In *KDD'08*, pages 990–998, 2008.

[12] Jianwen Wang, Xiaohua Hu, Xinhui Tu, and Tingting He. Author-conference topic-connection model for academic network search. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 2179–2183, New York, NY, USA, 2012. ACM.

[13] Z. Yang and B. D. Davison. Writing with style: Venue classification. In *2012 11th International Conference on Machine Learning and Applications*, volume 1, pages 250–255, Dec 2012.

[14] Zaihan Yang, Liangjie Hong, and Brian D. Davison. Academic network analysis: A joint topic modeling approach. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '13, pages 324–333, New York, NY, USA, 2013. ACM.

11 Appendix

11.1 Dataset Labels

ACM-Conf

Label	Num Training Examples	Num Dev/Test Examples
AAAI	2000	50
ACL	2000	50
ASPLOS	479	50
CCS	72	17
CHI	2000	50
CRYPTO	536	50
CVPR	2000	50
EMNLP	885	50
FOCS	1461	50
ICCV	1584	50
ICDM	1700	50
ICFP	97	24
ICML	1960	50
IEEESP	2000	50
MobiCom	429	50
MobiSys	363	50
NAACL	456	50
NSDI	231	50
OSDI	168	42
PLDI	320	50
PODS	283	50
POPL	622	50
SIGCOMM	2000	50
SIGGRAPH	2000	50
SIGIR	2000	50
SIGKDD	2000	50
SODA	1111	50
SOSP	365	50
STOC	1372	50
UbiComp	675	50
VLDB	2000	50

ACM-Field

Label	Abbreviation	Num Train Examples	Num Dev/Test Examples
Artificial Intelligence	AI	2000	50
Computer Vision	CV	2000	50
Databases	DB	2000	50
Graphics	GPH	2000	50
Human-Computer Interaction	HCI	2000	50
Information Retrieval	IR	2000	50
Machine Learning	ML	1960	50
Mobile Devices	MOB	1667	50
Networking	NET	2000	50
Natural-Language Processing	NLP	2000	50
Operating Systems	OS	1196	50
Programming Languages	PL	1185	50
Security	SEC	2000	50
Theory	TH	2000	50

ACM-Track

Label	Abbreviation	Num Train Examples	Num Test Examples
Artificial Intelligence	AI	2000	50
Graphics	GPH	2000	50
Human-Computer Interaction	HCI	2000	50
Information	INFO	2000	50
Networking	NET	2000	50
Systems	SYS	1659	50
Theory	TH	2000	50