CS224N : Predicting Category Tags by Using N-gram Models

HeeTae Jung Jonathan Hernandez YongRim Rhee

June 3, 2009

1 Introduction

The blogosphere is a highly populated medium in which people communicate and share their ideas. Amidst recent data explosion generated by bloggers, organization of these blogs has become an important task. Accordingly, a number of studies. e.g. [1, 2, 3], have been conducted to predict/suggest category tags for blog entries since tagging allows ranking and data organization to directly utilize inputs from end users. We implemented a category suggestion system using N-gram models. We also explore how category expanding and category clustering can affect the performance of the system. Our experimentation reveals that stochastic language models can effectively used to predict categories.

2 Data

We used the ICWSM weblog data for our experiments. ICWSM data is categorized into 14 different tier groups based on the ranking measured by an algorithm originated from TailRank. In the dataset, a blog entry is categorized into a higher tier group when it is measured to be more influential than other blog entries. Entries are categorized into 13 tier groups based on the ranking, and all the data that are not ranked are categorized into the non-tier group.

The ICWSM data covers August and September 2008, and is over 30GB compressed. Looking through the data, we noticed that many of the entries were not in English and that there were entries from Craigslist labeled as "classifieds." A lot of blog entries did not have any categories assigned to them. We decided we wanted to focus on blog entries in English that are actual blog entries as opposed to classifieds, and that they must have at least one category assigned to them. However, even limiting the data to this set of results is still a lot of data, and too much to process.

One of the major blog hosting websites is Wordpress. Choosing a major blog hosting site also solved another issue. The data contained a lot of very short entries from MySpace, but limiting it to Wordpress removes these. To limit our data, we decided to limit our results to only sites hosted on Wordpress.com. As this is one of the larger blog hosting sites out there, even limiting to this hosting site left us with too much data, so we decided to limit our data to tiers 1-3 and to blogs written on and between September 12th and September 18th. One of the reasons we chose these dates was that during this time period was when Lehman Brothers filed for bankruptcy(http://en.wikipedia.org/wiki/2008#September). Using these restrictions reduced the number of blog entries to around 20,000.

When running, we had five sets of data: one set was all 20,000 entries for training and 2000 for testing. The other sets were subsets of this first set - and contained only those blog entries which had a category that was popular. Popular categories are determined based on their number of occurrences in our data set.

data set	train data size	test data size
whole data	20,000	2,000
$threshold = 50$	9,000	1,000
$threshold = 100$	7,000	1,000
$threshold = 150$	6,000	1,000
$threshold = 200$	4,000	1,000

For example, when the threshold was set to 200, only the blog entries that had categories appearing more than and equal to 200 times were selected. That left us 5,000 blog entries in total, and we used 4,000 entries for training and 1,000 entries for testing.

3 Out Approach

3.1 N-gram Models

The Wordpress ICWSM blogs corpus contains, in its XML format, category tag representing the phrases the authors chose to tag their entries with. Using this tag and the sentences in the entry, we create and train 3 models for extracting categories. First in our baseline model we score the the likelihood of a category for word w for category $c = \frac{number\ of\ times\ word\ w\ seen\ by\ the\ category}{total\ number\ of\ words\ seen\ by\ the\ category}$. We then select a number of categories that returns the highest scores as a suggestion.

Though simple and effective, this approach does not take into account that each category could have distinct language models. The intuition behind this is that certain words may appear more often under certain categories. To build a language model for each of the categories, we build a unigram, bigram, and trigram languages model for each of the categories during training. We further implemented an interpolated model for all three of the N gram models. However, due to the limited computing resources available, we experiment only with the surface likelihood model, unigram and bigram model. In dealing with unseen words and sparcity of vocabulary introduced by the bigram model, we employ absolute discounting to smooth the ngram language models.

Evaluating the performance of our implementation, we pick the 6 highest scoring categories to compare to the testing data. When looking at the data, we saw that on average, each blog entry has about four categories, and we decided to let our algorithm assign up to 1.5 times the average, which is six.

3.2 Category Expanding

Bloggers give a number of category tags when they write blog entries. We examined the number of category tags that each blog entry has, and it turned out that each blog entry was assigned 3.9 category tags in average. This group of categories, usually called co-occurring categories, are assumed to jointly represent the content of the blog entry. Ciro, in [4], statistically studied the close relationship among category tags of the same blog entry and revealed that there is non-trivial relationship among them. The co-occurring tags have been used in tag clustering [5] and tag visualization system [6]. In our experiment, we explored if this co-occurrence information can be directly used to improve the performance of category tag prediction. Basically, we counted how often each category appears with other categories in the same blog postings by using the algorithm below, and the built map is directly used to expand the list of predicted categories.

```
for each blog b
  for all possible pairs of categories (c1, c2) of b
    increment the count by 1
  end for
end for
```

First, our N-gram model predicts the list of six categories which have the highest probabilities. Given this list, it is expanded by referring to the map built by the algorithm above. The list of categories can be expanded by either adding the mostly-related categories, i.e. the category which mostly co-occurred with the given category tag, or adding all the related categories, i.e. all the category tags which co-occurred with the given category tag.

3.3 Category Clustering

When we first got the data, we extracted the categories from some of the entries and looked at the results. We noticed that in some cases the same category would appear but would have a different capitalization. In addition, if a category was a person's name, one might have the first and last name, while another might just use the last name. For instance, we saw "Barack Obama," "Obama," "McCain," "John McCain" all as different categories. We wanted to investigate ways of automatically combining categories so that "Obama" and "Barack Obama" would be clustered as one category.

As we read in the training blog entries, we build a set of words for each category where these words are the words that appear in different blog entries with that category. To cluster categories, we want to compute the similarity of categories. The similarity is computed as the intersection of the words divided by the union of the words. Once we have these pairs of similar categories, we try to keep adding categories to this cluster that are similar to the ones already in this cluster, and when can add no more, call the result a cluster. To help cut down on the number of pairs of categories that are similar but should not be similar, we also develop a stop words list while training, and remove these words from the set of words for each category. To compute stop words, we keep a count of the number of blog entries that each word occurs in. Stop words are those words that occur in more than some threshold number of blog entries. We first tried some thresholds like 70%, but then tried smaller thresholds and the additional words did seem like good stop words. In the end, we used a threshold of 5%. There are perhaps a few words that should not be stop words that it finds, but in general most of the words do seem to be words that do not give much information.

Once we have these clusters, we compute new language models for each of the clusters by combining the language models(adding counters together) from each of the categories it contains, and remove from the individual category models. For the testing entries, we take each of its categories and see if it is in any clusters, if it is, we replace that category with all the clusters that that category is part of.

We noticed that there were a few categories that were often predicted for many entries and also similar to many other categories. One of these was "Life," probably since this encompasses many different ideas. If a category is more than a threshold similar to more than about 10 or 5 percent of all other categories, then we do not try to cluster this category with other categories since it seems to be a category that is very broad and would be clustered with many other categories. On the other hand, we could leave these in since this means that this category is one that is very broad and encompasses a lot of different and varied topics.

This does indeed often find clusters with categories that are related, but also finds some that at least to us do not seem so good. For instance, firebug, beta software, schphol, news and notes, chrome, treason, church unity, green bay packers, os x, is on one cluster that it finds, but also finds the cluster: firebug, web authoring tools, suzuki burgman, php, ajax, treason, software architecture, os x. The first cluster seems is mixed results - on the one hand, it does group firebug, chrome, beta software, and os x together, but then also groups a sports team with these, which does not seem correct. In the second case, looking up Suzuki Burgman gave me that it is a motorcycle. Looking at the entry with that label, it was mentioned in the entry but the entry was more of a tech entry. This is one example where now Suzuki Burgman will now often show up with technology/software labels just because happened to be mentioned in one tech blog entry. One problem may actually be that the article labeled "Suzuki Burgman" is quite long compared to many entries, which means that it has more words that could be shared with another category. After knowing why "Suzuku Burgman" shows up where it does, this cluster actually seems quite good: web authoring tools, suzuki burgman, gnu/linux. linux, debian.

4 Experiment Results

4.1 N-gram Models

The categories are taged at the author's discretion. Although this makes it very difficult to evaluate the quality of our system, we can take a glimpse to to scoring for each of our model and see how each model scores the example entry.

```
Title: I Love This!
Author's categories: [humor, politics]

the political theater of the past few weeks has been absolutely
hilarious. democrats and the rest of the left (*cough* the media
 *cough*) have been frothing at the mouth to attack sarah palin
so much that they entrely fail to look at how well their worshipped
candidate stands up against such attacks. the latest examples?
heavy criticism of sarah palin's stance that we should allow
ukraine and georgia into nato, since that would increase tensions
with russia. of course, obama and biden hold the exact same stance
as palin on this issue. there's also continuing attempts to point
out that sarah palin wasn't initially against the bridge to nowhere.
which is true (and which i mentioned before).  however, she is the
one responsible for killing the bridge - and not only that, but obama
and biden both explicitly voted not to kill the bridge measure, a vote
which even markos moulitsas said showed they had "zero credibility
on issues of fiscal responsibility".


baseline suggestions
        [politics] : -292.350063122478
            [news] : -447.4144014310147
            [life] : -451.65631786622214
     [sarah palin] : -471.1488160057377
           [obama] : -484.1979023164756
     [john mccain] : -494.72596911062345

smoothed unigram suggestions
        [politics] : -1867.5046219015048
     [john mccain] : -2191.4145557836937
     [sarah palin] : -2418.340810458445
    [barack obama] : -2432.6114131426
            [news] : -2465.788195845551
           [obama] : -2549.4630790414576

smoothed bigram suggestions
        [politics] : -8133.013071141855
```

```
  [john mccain] : -9855.08272477071
  [sarah palin] : -9981.510284574928
         [news] : -10635.699420672248
        [obama] : -10807.042318885264
[barack obama] : -10888.055112434698
```

All three models correctly guess that 'politics' is the highest scoring category. Interestingly, though John McCain is never mentioned all three models suggest John McCain as a possible tag. Further more the unigram and the bigram models both suggest John McCain as its 2nd highest ranking category. This suggests that in both modeling unigram and bigram language models, John McCain was used as a tag in entries that used similar unigram and bigram words. We can see the significant improvement in the unigram and the bigram models since we don't see the tag "life" suggested in either models even though it is one of the most commonly used tags in the blogosphere. In fact, in both unigram and bigram models for this example text, "life" was one of the lower ranking categories suggested by the system.

The following table shows some of the results by using different data set and different models; s-unigram indicates smoothed unigram model and s-bigram indicates smoothed bigram model. Category expansion, however, added mostly-related categories only. E-precision indicates expanded precision and e-recall indicates expanded recall; used to reduce the size of the table.

data set	precision	recall	e-precision	e-recall
whole data with baseline	0.0632	0.0342	0.1573	0.0051
whole data with s-unigram	0.0868	0.0441	0.0953	0.0356
whole data with s-bigram	0.0892	0.0432	0.1799	0.0071
threshold = 50 with baseline	0.2078	0.0715	0.2344	0.0569
threshold = 50 with s-unigram	0.2632	0.0847	0.2905	0.0691
threshold = 50 with s-bigram	0.2518	0.0837	0.2729	0.0676
threshold = 100 with baseline	0.2965	0.0902	0.3365	0.0717
threshold = 100 with s-unigram	0.3710	0.1060	0.4117	0.0868
threshold = 100 with s-bigram	0.3545	0.1045	0.3883	0.0850
threshold = 150 with baseline	0.3856	0.1030	0.4415	0.0814
threshold = 150 with s-unigram	0.4500	0.1183	0.5156	0.0972
threshold = 150 with s-bigram	0.4478	0.1170	0.4926	0.0951
threshold = 200 with baseline	0.4952	0.1245	0.5432	0.1003
threshold = 200 with s-unigram	0.5821	0.1455	0.6336	0.1226
threshold = 200 with s-bigram	0.5573	0.1412	0.6004	0.1192

4.2 Category Expanding

We witnessed significant improvement in precision when we adopted category expansion. As can be seen in the table, we were able to increase precision while not losing too much in the recall when we expanded by adding mostly-related categories. The following table shows you the result that we got using our smoothed unigram model.

data set	precision	recall	expanded precision	expanded recall
whole data	0.0868	0.0441	0.0953	0.0356
threshold = 50	0.2632	0.0847	0.2905	0.0691
threshold = 100	0.3710	0.1060	0.4117	0.0868
threshold = 150	0.4500	0.1183	0.5156	0.0972
threshold = 200	0.5821	0.1455	0.6336	0.1226

Let us look at the experiment that we ran on the subset of blog entries which had categories that appeared more than or equal to 200 times. A predicted category "politics" was expanded to "politics, sarah palin," meaning that bloggers wrote about "sarah palin" when they wrote about "politics" in the time period we picked.

data set	precision	recall	expanded precision	expanded recall
whole data	0.0868	0.0441	0.0953	0.0356
threshold = 50	0.2632	0.0847	0.6706	0.0187
threshold = 100	0.3710	0.1060	0.8359	0.0287
threshold = 150	0.4500	0.1183	0.9495	0.0377
threshold = 200	0.5821	0.1455	0.9680	0.0491

When we expanded the same category "politics" by adding all the related categories, our system returned the list of categories "sarah palin, news, john mccain, barack obama, media, obama, ...". By adopting these fully expanded list of category tags, we were able to boost the performance even higher. The initial average precision was 0.58, and this got increased to 0.63 when we added just one mostly related categories to the predicted list. The precision jumped up to 0.97 by adding all the related categories to the predicted one. In that case, we observed significant drop in recall. Even considering the apparent lost in recall, we believe that the result reveals that there are apparent pattern when bloggers give category tags to their blog entries and confirms the Ciro's research [4].

4.3 Category Clustering

Adopting category clustering did not actually improve our results very much. In some cases, it would offer a very small improvement over either recall or precision, but would then decrease in the other. The following table shows some of the examples; the first two columns are the performance of smoothed unigram model without adopting category clustring, and the last two columns are the one with adopting category clustering. When the category clustering was tested on the whole data, it showed slight depreciation in precision. However, when tested on the data with the threshold one hundred, it improved both precision and recall.

data set	precision	recall	precision	recall
whole data	0.0868	0.0441	0.0833	0.0445
threshold = 100	0.3710	0.1060	0.4252	0.1151

One of the things that makes assigning categories hard is that each person may use categories slightly different and there are thousands of unique categories. One issue may be that since we only allow our algorithm to assign 6 categories, when we use clusters, since a category may be in several clusters, we perhaps should increase the number allowed to be assigned, perhaps adjust this dynamically. Another improvement might be to take these clusters, and see if any of these new clusters can be combined to get larger clusters, which should start to represent larger and larger topics perhaps. One entry titled "Mistress ?Rules? North Korea?" has the categories "Daily doilies" and "kim jong-il's mistress". Our algorithm does not get these categories, but does predict "politics," "history," "news" and a cluster of linda sanches, John McCain, treason, Barack Obama all of which seem reasonable categories given the title, especially "politics," but this is not one that the author assigned to the blog. We do not have a way to take this into account - the fact that our algorithm may find a category or cluster that is close or could be applied to the blog but was not. Or for instance, given the the title, it seems likely that the category "music" could apply to "Apple iTunes Genius" but the only category it is labeled with is "technology."

4.4 Blog Entry Lengths

When we saw that a lot of entries had very few words, we thought that perhaps by disregarding these during training, we might get better results. However, in fact the opposite occurred. We happened to try running also on only the ones we wanted to throw out, and were surprised to see how relatively good the results were. Even the short entries have a lot of information that is useful for the models for the categories. However, in order to be able to do a good comparison of results, we used the same testing set, and thus were also testing on entries that may be quite short as well. Thus, one might want to compare results when only testing and training on larger entries.

The following table shows the example of the experiments. The smoothed unigram model was used to train and test, and mostly-related categories were added at the category expansion stage.

data set	precision	recall	precision	recall
whole data with more than 78 words	0.0759	0.0410	0.1669	0.0062
whole data with less than 78 words	0.0814	0.0377	0.1753	0.0056

We looked at the number of words in each blog entry. The average length across our test and train sets was only 78 words, which is quite small. Then there was one blog entry with over 8000 words. Some blogs do not have as much text as others but rather have more multimedia such as photos or videos. In these, there is much less text. We thought that we might be able to improve our results by ignoring entries with less than the average number of words, but instead, recall and precision were actually worse. One reason may be that there were so many entries with small numbers of words and thus we are losing a lot of information. Ideally, we would be able to remove blogs that are photoblogs, or ones that mainly have videos, etc, from both our training and test set. To better predict labels for these, one would want to use other ways of extracting data from other media. If a human only saw the text from a photoblog and was asked to classify it, one would likely have a much harder time than if we could have the context of the media.

5 Discussion and Conclusion

In some blog entries, the entry is very focused on one subject only, and thus the categories tend to match this subject. However, in other cases, an entry might be say more like an entry in a diary. Thus it may be about daily activities - school, work, home, etc, but may also mention some political news perhaps, and thus might have a tag or tags related to politics or news or the specific news. Another step might be to detect topic shifts in blog entries, and thus try to determine which categories apply to which particular topic during training, and then for testing, split entries into the different topics and assign categories to the different parts of the entry. One of the issues that makes this area difficult is that the categories that users assign to their blogs are not standardized - "Obama" and "Barack Obama" - and also that bloggers may tag their blog with a few categories, but perhaps there are several more categories that equally apply in addition, but we cannot measure this if our algorithm finds these other categories, as currently these are treated as incorrect. The other issue is that there are so many potential categories to test, and can only assign a few. Perhaps by using larger and thus fewer clusters, might help. From above, the entry labeled "Firebug" is actually a relatively long entry, which may make it easier to be grouped with other categories for better or for worse. One may want to try adjusting thresholds for determining similarities for clustering perhaps based on the amount of data we have about those categories - perhaps the number of words and also number of blogs with that category. We thought that clustering would get all the tags about Gov. Sarah Palin, for instance, together into one cluster, but found that this was not as easy as it seemed using similarities of categories, since perhaps in one case someone just mentioned some news about her, while another was all about the election, while another could be about Alaska. One way to help combine might be to actually look at the titles and see if words in one category title appear in another category title, and if so and if the categories are somewhat similar, then just go ahead and merge perhaps.

References

[1] Zhichen Xu, Yun Fu, Jianchang Mao and Difu Su, *Towards the Semantic Web: Collaborative Tag Suggestions*, WWW '06

[2] Gilad Mishne,*AutoTag: A Collaborative Approach to Automated Tag Assignment for Weblog Posts*, WWW '06

[3] Sanjay Sood, Sara Owsley, Kristian Hammond and Larry Birnbaum, *TagAssist: Automatic Tag Suggestion for Blog Posts*, ICWSM '07

[4] Ciro Cattuto, Vittorio Loreto and Luciano Pietronero, *Semiotic Dynimcs and Collaborative Tagging*, PNAS '07

[5] G. Begelman, P. Keller and F. Smadja, *Automated Tag Clustering: Improving Search and Exploration in the Tag Space*, WWW '06

[6] Yusef Hassan Montero and Victor Herrero-Solana, *Improving Tag-Clouds as Visual Information Retrieval Interfaces*, InSciT '06